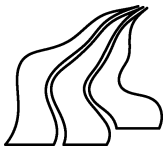
Aalborg University

Department of Computer Science



Title:
Adaptive Bayesian Networks for Zero-Sum Games

Project period:
1/2-2000 – 31/5-2000

Project group: DAT6, NOVI
Thomas Jørgensen

Supervisor:
Professor Finn V. Jensen

Total number of pages: 78

Number of reports printed: 5

Abstract:

In this masters thesis we focus on the use of Bayesian networks for solving finite two-person zero-sum games with one decision. We look into the details of an iterative solution procedure suggested by George W. Brown in 1949 and verifies it formally as well as in practice. We show that the solutions found by Brown's procedure are *Nash equilibria* of the games we use as testbed.

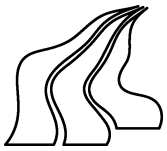
We show that we can use the principles from Brown's solution procedure to create adaptive Bayesian networks capable of solving a game by letting intelligent agents based on the adaptive networks face each other. Their distributions will during the series of games converge against the *Nash equilibrium* for the game.

Finally we show that Bayesian networks are capable of solving more complex games with more than one decision, by using training methods of the same principle as Brown's solution procedure.

Copyright ©2000, DAT6, Jørgensen

Aalborg Universitet

Institut for Datalogi



Titel:
Adaptive Bayesian
Networks for
Zero-Sum Games

Projekt periode:
1/2-2000 – 31/5-2000

Projektgruppe: DAT6, NOVI
Thomas Jørgensen

Vejleder:
Professor Finn V. Jensen

Totalt antal sider: 78

Antal rapporter printet: 5

Abstract:

I dette speciale fokuserer vi på brugen af Bayesianske netværk til løsning af endelige to-personers nulsumsspil med med een beslutning. Vi går i detaljer med en iterativ løsningsmetode foreslået af George W. Brown i 1949 og viser at den virker såvel formelt som i praksis. Vi efterviser desuden at de løsninger vi finder med Browns løsningsmetode er *Nash ligevægte* for de spil de løser.
Vi viser at vi kan bruge principperne fra Browns løsningmetode til at lave adaptive Bayesianske netværk der kan løse et spil ved at lade intelligent agenter baserede på de adaptive netværk møde hinanden i en serie af spil. Deres distributioner vil da konvergere mod *Nash ligevægten* for spillet.
Vi slutter af med at vise at Bayesianske netværk kan løse mere komplicerede spil med mere end en beslutning ved at bruge træningsmetoder af samme princip som Brown's løsningsprocedure.

Copyright ©2000, DAT6, Jørgensen

Contents

Preface

This thesis is the result of the work done by group DAT6-NOVI in the spring of 2000. We continue the work from DAT5 as described in [Jørgensen, 2000], where we made some initial studies on the use of Bayesian networks in the area of game theory.

The main subject of this report is solution of finite two-person zero-sum games with adaptive Bayesian networks. As usually when working with game theory, we describe the first player using male pronouns and the second player using female pronouns. If a player is referred without role, we shall use male pronouns.

The notation used in this thesis is mainly based on [Jensen, 1996] when concerning Bayesian networks and [Robinson, 1951] when describing the suggested solution procedure for the games.

This thesis has been typeset with LaTeX.

We would like to thank HUGIN Expert A/S for providing us with a version of the HUGIN Java API during the project period.

Thomas Jørgensen

Chapter 1

Introduction

The purpose of this thesis is to continue the work from [Jørgensen, 2000] where we concluded that if we let intelligent agents based on adaptive Bayesian networks play against each other, they converge against the same probability distributions. From these distributions we can read out *randomized strategies* describing the behavior of the agents during the series of games. In [Jørgensen, 2000] we argued that the final strategies for the game of spoofing "seemed" to reflect an intelligent and rational behavior.

In this thesis we would like to prove that intelligent agents based on adaptive Bayesian networks actually converge against a set of randomized strategies that is a solution of the game they are playing. Such a solution is what we in [Jørgensen, 2000] described as a *Nash equilibrium*. To be able to actually prove that the randomized strategies we end up with are a Nash equilibrium we start out with some very simple games where we can pre-compute the solutions. The games that are the subject of this thesis are finite two-person zero-sum games.

As a theoretic foundation for solution of such simple games we use the work of George W. Brown and Julia Robinson as described in [Robinson, 1951]. In this article Robinson is describing and proving an iterative solution procedure for finite two-person zero-sum games. This procedure is initially suggested by George W. Brown in 1949 in some unpublished work. In Chapter 2 we introduce some general game theoretic concepts to ensure a clear understanding of the theory behind zero-sum games, and we describe in detail the solution procedure and the proof hereof.

In Chapter 3 we try to verify the iterative solution procedure by testing it on a few simple games, before we in Chapter 4 introduce an integration of the

solution procedure into Bayesian networks. We shall also verify that these networks are capable of solving the simple games we consider in this work.

Finally, in Chapter 5 we move away from the type of games covered by the iterative solution procedure suggested by Brown, to see if we can solve more complex games with our new representation in adaptive Bayesian networks. The results are summed up in Chapter 6 and a discussion of future work is included in Chapter 7.

Chapter 2

Iterative Solution of Finite Two-Person Zero-Sum Games

Much of the early work in the area of Game Theory was done on two-person zero-sum games since those are the games that are easiest to model mathematically. As work proceeded, the simplifying assumption of zero-sum, meaning that ones gain is always the opponents loss, was found to prohibit the modeling of more complex and realistic games, and the focus started turning to non-zero-sum games. However, the theory of zero-sum games still contains a lot of interesting aspects as we shall discover in the following sections.

One of the most active researchers in zero-sum games was *John von Neumann*, which is also the father of the widely referred *minimax theorem of game theory*. As well as this theorem is widely referred it is also widely rewritten and reproved, in this thesis we shall adopt the notion from recent works in probability- and decision theory together with the most suitable notion for the main task of this chapter.
Back in 1949, *George W. Brown* suggested a method for an iterative solution of finite two-person zero-sum games with one decision. In 1951 *Julia Robinson*[Robinson, 1951] proved the validity of the procedure suggested by Brown. The purpose of this chapter is to highlight the results from Brown and Robinson, and to do so we start looking at some of the earlier results to ensure a clear understanding of the concepts. The definitions are given using games with only pure strategies, but they can easily be extended to include *randomized play*[Myerson, 1991], however, doing so would make them less suitable to serve as a gentle introduction. *Randomized play* will be introduced later on.

To represent the games we use the *strategic form*. To define a game in

strategic form we need to define a set of players, the set of *strategies* available to the players and the pay-off they gain from the various *strategies*. In the context of game theory a *strategy* is defined as follows

Definition 1 (Strategy)
A strategy for a player is a prescription for what the player must do in any possible decision situation that can happen.

and a *strategic form game* is defined as

Definition 2 (Strategic Form Game)
A strategic form game is any Γ *of the form*

$$\Gamma = (N, (C_p)_{p \in N}, (u_p)_{p \in N}), \text{ where}$$

N *is the set of players,* $N \neq \emptyset$
C_p *is the set of strategies available to player* p*, where* $\forall p \in N, C_p \neq \emptyset$
$u_p : C \to R$ *is the utility pay-off function for player* p*, where* $C = \{C_p\}_{p \in N}$

In our case where we consider only two-person zero-sum games we can use the slightly simpler Definition 3.

Definition 3 (Two Person Zero-Sum Game)
A two person zero-sum game in strategic form is any Γ *of the form*

$$\Gamma = (\{1, 2\}, C_1, C_2, u_1, u_2)$$

such that

$$u_2(i, j) = -u_1(i, j), \ \forall i \in C_1, \ \forall j \in C_2$$

If we assume that C_1 and C_2 are finite sets, then in two-person games, it is convenient to represent the utility function as a pay-off matrix

$$A = a_{ij}, \ \forall i \in C_1, \ \forall j \in C_2$$

If Player1 chooses the i^{th} row and Player2 simultaneously chooses the j^{th} column, then Player2 pays a_{ij} to Player1.
The matrix representation of the utility function will work with randomized- as well as with pure strategies. If we are using only pure strategies the utility matrix alone can be used for finding *optimal strategies*. When speaking of an *optimal strategy* we are referring to optimality in terms of expected winnings, thus an optimal strategy is optimizing your winnings or minimizing your loss. When using only pure strategies Definition 4 is used to define optimal strategies.

Definition 4 (Saddle Point)
The strategy set (i', j') *is called a saddle point for* Γ *iff*

$$a_{ij'} \leq a_{i'j'} \leq a_{i'j}, \ \forall i \in C_1 \ \forall j \in C_2$$

The value $a_{i'j'}$ *is called the saddle value, and* i' *and* j' *are called optimal strategies.*

In other words, if Player1 plays the strategy i' he is guaranteed to win at least *the* saddle value $a_{i'j'}$ and Player2 following the strategy j' is guaranteed not to loose more. If one of the players plays another strategy he can never be better off, and most likely he will be worse off. Thus, i' and j' are optimal strategies in the way that they guarantee the maximal gain against an intelligent, rational opponent. Which strategies are the best against suboptimal opponents shall be unsaid.
In the previous section we italicized *the* saddle value. Note that there can be more than one saddle point in the same game. In Figure 2.1 say that (i', j') and (i'', j'') are saddle points, thus from Definition 4 we have that v' is the largest element in the row labelled i' and furthermore the smallest element in the column labelled j'. Similarly for i'' and j''.

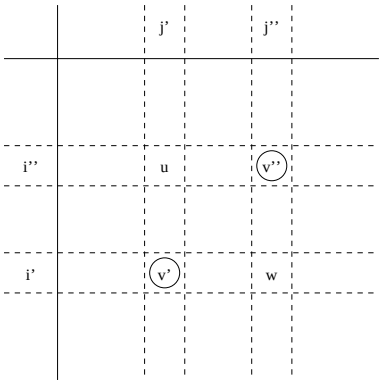


Figure 2.1: *Value equivalence between several saddle points*

Thus we can write

$$w \leq v' \leq u \text{ and } u \leq v'' \leq w$$

and then we have that if we have more than one saddle point in a game the saddle value is the same for all of them.

2.1 Rational Reasoning

Since we are assuming our opponent to be intelligent and rational we might as well expect him to be playing his optimal strategies. In this section we shall see how players should reason in search for optimal strategies.
Remember the basic property of zero-sum games

$$u_1(i,\ j) = -u_2(i,\ j) \tag{2.1}$$

With this in mind it is easy to see that the task of maximizing your winnings is equivalent to minimizing your opponents winnings. Since we expect rational and intelligent opponents, a player should always assume that the opponent plays the best possible strategy against optimal play. In other words, Player1 should assume that Player2 is solving the problem

$$\underset{j \in C_2}{min}\ a_{ij} \tag{2.2}$$

Thus, Player1 must solve the problem

$$V_1 = \underset{i \in C_1}{max}\ (\underset{j \in C_2}{min}\ a_{ij}) \tag{2.3}$$

where V_1 is Player1's minimum winnings from the game. Similarly Player2 must minimize her loss, so she should solve the problem

$$V_2 = \underset{j \in C_2}{min}\ (\underset{i \in C_1}{max}\ a_{ij}) \tag{2.4}$$

where V_2 is Player2's maximum loss.
Since V_2 is the maximum possible loss of Player2 and the game is zero-sum, it seems reasonable to state that no matter what strategy Player1 is using he can not possible win more than V_2, thus

$$V_1 \leq V_2 \tag{2.5}$$

This leads to the following theorem

THEOREM 1
For finite two-person zero-sum games (as well as for any other matrix) the following inequality is true

$$\underset{i \in C_1}{max}\ (\underset{j \in C_2}{min}\ a_{ij}) \leq \underset{j \in C_2}{min}\ (\underset{i \in C_1}{max}\ a_{ij})$$

PROOF:
For static $i' \in C_1$ and $j' \in C_2$ it is easy to see that

$$a_{i'j'} \geq \underset{j \in C_2}{min}\ a_{i'j}, \forall i' \in C_1, \forall j' \in C_2$$

By taking max on both sides we get

$$\underset{i \in C_1}{max}\ a_{ij'} \geq \underset{i \in C_1}{max}\ (\underset{j \in C_2}{min}\ a_{ij}), \forall j' \in C_2$$

From here Theorem 1 follows.

QED

The way we got to Theorem 1 was by arguing how a player should behave to ensure optimal play, therefore it seems reasonable to assume that there is a relation with saddle points from Definition 4, actually the following is true

THEOREM 2
(i', j') is a saddle point for $\Gamma = (\{1, 2\}, C_1, C_2, u_1, u_2)$ iff

$$\underset{j \in C_2}{min}\ a_{i'j} = \underset{i \in C_1}{max}\ (\underset{j \in C_2}{min}\ a_{ij}) = \underset{j \in C_2}{min}\ (\underset{i \in C_1}{max}\ a_{ij}) = \underset{i \in C_1}{max}\ a_{ij'}$$

PROOF:
Assume that (i', j') is a saddle point for Γ, then

$$a_{ij'} \leq a_{i'j'} \leq a_{i'j} \forall i \in C_1, \forall j \in C_2$$

and it follows that

$$\underset{i \in C_1}{max}\ a_{ij'} \leq a_{i'j'} \leq \underset{j \in C_2}{min}\ a_{i'j}$$

and thus

$$\underset{j \in C_2}{min}\ (\underset{i \in C_1}{max}\ a_{ij}) \leq \underset{i \in C_1}{max}\ a_{ij'} \leq a_{i'j'} \leq \underset{j \in C_2}{min}\ a_{i'j} \leq \underset{i \in C_1}{max}\ (\underset{j \in C_2}{min}\ a_{ij})$$

combining this with Theorem 1 we get Theorem 2.
Now we assume that Theorem 2 is true, and we can write

$$a_{i'j'} \leq \underset{i \in C_1}{max}\ a_{ij'} = \underset{j \in C_2}{min}\ a_{i'j} \leq a_{i'j'}$$

and now we have that

$$\underset{i \in C_1}{max}\ a_{ij'} = a_{i'j'} = \underset{j \in C_2}{min}\ a_{i'j}$$

From Definition 4 we see that $(i',\ j')$ is a saddle point.

QED

This concludes the gentle introduction to the area of two-person finite zero-sum games, and it is time to complicate things and introduce randomized strategies.

2.2 Equilibria in Randomized Strategies

With the introduction of *randomized strategies* it is possible for equilibria to occur in games that have no equilibria in pure strategies. A *randomized strategy* for Player p is any probability distribution over C_p, and we let $\Delta(C_p)$ denote the set of all possible randomized strategies for player p.
A *randomized strategy profile* is any vector that specifies one randomized strategy for each player, so $\times_{p \in N} \Delta(C_p)$ is the set of all randomized strategy profiles.
To justify the search for equilibria in finite two-person zero-sum games, we give, without the proof, the *general existence theorem* introduced by John Nash in 1951[Myerson, 1991]

THEOREM 3
Given any finite game Γ *in strategic form, there exists at least one equilibrium in* $\times_{p \in N} \Delta(C_p)$

With this in mind we are ready to start looking at Brown's procedure for finding such equilibria. Remember that we represent our utility function as a matrix

$$A = a_{ij}, \forall i \in C_1, \forall j \in C_2$$

We let Player1 play the i^{th} row with probability x_i and similarly Player2 play the j^{th} column with probability y_j, where $x_i \geq 0, y_i \geq 0, \sum x_i = 1$ and $\sum y_i = 1$. We can calculate the expected utility of Player1 as

$$EU = \sum_i \sum_j a_{ij} x_i y_j \tag{2.6}$$

We have that

$$\min_j \sum_i a_{ij} x_i \leq \sum_i \sum_j a_{ij} x_i y_j \leq \max_i \sum_j a_{ij} y_j \tag{2.7}$$

to see why this is so, consider the middle term as

$$\sum_j y_j \sum_i a_{ij} x_i \tag{2.8}$$

Since $\sum_j y_j = 1$ we can consider 2.8 as a weighted average over $\sum_i a_{ij} x_i$, and therefore it is true that

$$\min_j \sum_i a_{ij} x_i \leq \sum_j y_j \sum_i a_{ij} x_i$$

Similarly we can write the middle term of 2.7 as

$$\sum_i x_i \sum_j a_{ij} y_j$$

which is a weighted average over $\sum_j a_{ij} y_j$ and thus

$$\sum_i x_i \sum_j a_{ij} y_j \leq \max_i \sum_j a_{ij} y_j$$

From 2.7 it naturally follows that

$$\min_j \sum_i a_{ij} x_i \leq \max_i \sum_j a_{ij} y_j \tag{2.9}$$

Looking at 2.9 we see that if the equality holds, it is somehow similar to Theorem 2, stating that

$$\min_{j \in C_2} a_{i'j} = \max_{i \in C_1} a_{ij'}$$

The difference is that 2.9 includes randomized strategies which is not the case in Theorem 2 since the latter was kept simple to ensure a clear understanding of the concepts. However, they are both defining optimal strategies, Theorem 2 is defining optimal pure strategies in opposition to 2.9 which is defining optimal randomized strategies over the pure strategies.
In fact it is true, that even though there may be no equilibria in pure strategies, the equality in 2.9 holds for some set of probabilities $X = (x_1, ..., x_m)$ and $Y = (y_1, ..., y_n)$. This result is widely known as *the minimax theorem of game theory* and can be found in [von Neumann and Morgenstern, 1944].
Such a pair of probability sets, (X, Y) is called a solution or a *Nash Equilibrium* of the game and *the value*, v, of the game is defined as

$$v = \min_j \sum_i a_{ij} x_i = \max_i \sum_j a_{ij} y_j \tag{2.10}$$

As in pure strategies, this value is the same for all equilibria. To see why this is so, consider the following
From 2.9 we have that

$$\min_j \sum_i a_{ij} x_i \leq \max_i \sum_j a_{ij} y_j$$

which is true for all probability sets (X', Y')

Say we have two Nash equilibria of the same game, call them (S, T) and (U, W). Then we have from 2.10 that

$$\min_j \sum_i a_{ij} s_i = \max_i \sum_j a_{ij} t_j = v$$

and

$$\min_j \sum_i a_{ij} u_i = \max_i \sum_j a_{ij} w_j = v'$$

Now if we consider (S, T) and replace the T with W we know that

$$\min_j \sum_i a_{ij} s_i \leq \max_i \sum_j a_{ij} w_j$$

and get

$$v \leq v'$$

Similarly if we consider (U, W) and replace the W with T we get that

$$\min_j \sum_i a_{ij} u_i \leq \max_i \sum_j a_{ij} t_j$$

and therefore have that

$$v' \leq v$$

We now see that $v = v'$
In terms of rows and columns in the pay-off matrix, equation 2.9 can be viewed as

$$min \sum_i A_{i.} x_i \leq max \sum_j A_{.j} y_j \tag{2.11}$$

Where $A_{.i}$ is the i^{th} row of A and $A_{j.}$ is the j^{th} column of A
With the basic concepts defined we are now ready to move on to Brown's theorem on *vector systems*.

2.3 Brown's Theorem

Brown's procedure for solving games is based on recursive manipulation of vectors resulting in what is referred to as a *vector system* for A. A *vector system* is defined as

Definition 5 (Vector System)
A system (U, V) consisting of a sequence of n-dimensional vectors $U(0), U(1), ...$ *and a sequence of m-dimensional vectors* $V(0), V(1), ...$ *is a vector system for A iff*

$$minU(0) = maxV(0)$$

and

$$U(t+1) = U(t) + A_{i\cdot}, \qquad V(t+1) = V(t) + A_{\cdot j},$$

where i and j satisfy the conditions

$$v_i(t) = maxV(t), \qquad u_j(t) = minU(t)$$

From Definition 5 we can see that it is possible to recursively form a vector system given any initial vectors $U(0)$ and $V(0)$. In [Robinson, 1951] the case

$$U(0) = V(0) = \overrightarrow{0}$$

is considered as a special case since the definition is valid for all initial vectors. However, since we are to use the procedure only as a way of solving finite two-person zero-sum games, we shall not consider cases where $U(0) \neq V(0) \neq \overrightarrow{0}$. We can now consider $\frac{U(t)}{t}$ and $\frac{V(t)}{t}$ as a weighted average of the rows and columns respectively, where the weighting factors are the number of times the row or column has been chosen divided by the number of iterations. Formally

$$\frac{U(t)}{t} = \frac{n_i}{t} A_{i\cdot} \ ,$$

where n_i is the number of times the i^{th} row has been chosen
Similarly

$$\frac{V(t)}{t} = \frac{m_j}{t} A_{\cdot j} \ ,$$

where m_j is the number of times the j^{th} column has been chosen
Since $\frac{U(t)}{t}$ is a weighted average over the rows of A we have that

$$\frac{minU(t)}{t} \leq \max_i \sum_j a_{ij} y_j \tag{2.12}$$

Similarly since $\frac{V(t)}{t}$ is a weighted average over the columns of A we have that

$$\frac{maxV(t)}{t} \geq \min_j \sum_i a_{ij} x_i \tag{2.13}$$

Combining 2.10, 2.12 and 2.13 we get for every t and t'

$$\frac{minU(t)}{t} \leq v \leq \frac{maxV(t')}{t'} \tag{2.14}$$

Brown's result states that if for some t and t' it is true that

$$\frac{minU(t)}{t} = \frac{maxV(t')}{t'} = v$$

we have a solution of the game. The solution, which is an optimal randomized strategy, can be read out as the number of times the rows and columns were chosen divided by the total number of iterations.
Even if we never find an exact solution Brown states the following theorem which is the main result of his work

THEOREM 4
If (U, V) is a vector system for A, then

$$\lim_{t\to\infty} \frac{minU(t)}{t} = \lim_{t\to\infty} \frac{maxV(t)}{t} = v$$

The proof of Theorem 4 will be divided into 4 lemmas.

LEMMA 1
If (U, V) is a vector system for A, then

$$\lim_{t\to\infty} \; inf \frac{maxV(t) - minU(t)}{t} \geq 0$$

PROOF:
Since $V(t)$ is constructed as a weighted average over the columns of A and we made the assumption that $U(0) = V(0) = \overrightarrow{0}$, we have that

$$V(t) = t\sum_j y_j A_{\cdot j}, \text{ where } \sum y_j = 1 \text{ and } y_j \geq 0, \forall j$$

Similarly for $U(t)$

$$U(t) = t\sum_i x_i A_{i\cdot}, \text{ where } \sum x_i = 1 \text{ and } x_i \geq 0, \forall i$$

However, Theorem 4 is given for any vector system so we might have a case where $U(0) \neq V(0) \neq \overrightarrow{0}$ and therefore we have to consider $U(t)$ and $V(t)$ as follows

$$V(t) = V(0) + t\sum_j y_j A_{\cdot j}, \text{ where } \sum y_j = 1 \text{ and } y_j \geq 0, \forall j$$

$$U(t) = U(0) + t\sum_i x_i A_{i\cdot}, \text{ where } \sum x_i = 1 \text{ and } x_i \geq 0, \forall i$$

By choosing the minimum value of $V(0)$ we are sure that the following inequality is true

$$maxV(t) \geq minV(0) + t\ max\sum_j y_j A_{\cdot j}$$

In the same way we get that

$$minU(t) \leq maxU(0) + t\ min\sum_i x_i A_{i\cdot}$$

Hence,

$$maxV(t) - minU(t) \geq minV(0) - maxU(0) + t\left(max\sum_j y_j A_{\cdot j} - min\sum_i x_i A_{i\cdot}\right)$$

As

$$\lim_{t\to\infty} \frac{minU(0) - maxV(0)}{t} = 0$$

we get that

$$\frac{maxV(t) - minU(t)}{t} \geq max\ \sum_j y_j A_{\cdot j} - min\ \sum_i x_i A_{i\cdot}$$

From 2.11 we get that

$$\frac{maxV(t) - minU(t)}{t} \geq 0$$

which yields the lemma.

QED

For the next Lemmas we need to introduce the concept of eligibility.

Definition 6 (Eligibility)
If (U, V) is a vector system for A, we say that the i^{th} row is eligible in the interval (t, t') iff there exists a t_1 such that

$$t \leq t_1 \leq t'$$

and

$$v_i(t_1) = maxV(t_1)$$

In the same way we say that the j^{th} column is eligible in the interval (t, t') iff there exists a t_2 such that

$$t \leq t_2 \leq t'$$

and

$$u_j(t_2) = minU(t_2)$$

In words, an eligible row or column is one that can be chosen in the given interval during the iterative solution procedure. With this defined we are ready to move on to the next lemma.

LEMMA 2
If (U, V) is a vector system for matrix A and all the rows and columns of A are eligible in the interval (s, s+t) we have that

$$maxU(s+t) - minU(s+t) \leq 2at$$

and

$$maxV(s+t) - minV(s+t) \leq 2at$$

where

$$a = max_{i,\ j}|a_{ij}|$$

PROOF:
Choose j such that

$$u_j(s+t) = maxU(s+t)$$

and as j is eligible we can choose t' such that $s \leq t' \leq s+t$ and

$$u_j(t') = minU(t')$$

We know that a is the maximum possible change per iteration, and therefore we have that at is the maximum change in t iterations.
Thus, because we chose t' between s and $s+t$, we know that the difference between $u_j(s+t)$ and $u_j(t')$ can at most be at, we have that

$$u_j(s+t) \leq u_j(t') + at = minU(t') + at$$

and from the way we chose j we now have that

$$maxU(s+t) \leq minU(t') + at$$

which can also be written as

$$minU(t') \geq maxU(s+t) - at \tag{2.15}$$

Again, by looking at the way we chose t' and the maximum difference we can reach in t iterations, we get

$$minU(s+t) \geq minU(t') - at \tag{2.16}$$

By insertion of 2.15 in 2.16 we get

$$minU(s+t) \geq maxU(s+t) - 2at$$

which we can write as

$$maxU(s+t) - minU(s+t) \leq 2at$$

In the same way it can be shown that

$$maxV(s+t) - minV(s+t) \leq 2at$$

QED

LEMMA 3
If (U, V) is a vector system for matrix A, and all the rows and columns of A are eligible in the interval (s, s+t) it is true that

$$maxV(s+t) - minU(s+t) \leq 4at$$

PROOF:
From Lemma 2 we have that

$$(maxU(s+t) - minU(s+t) + (maxV(s+t) - minV(s+t)) \leq 4at$$

This can as well be written as

$$maxV(s+t) - minU(s+t) \leq 4at - maxU(s+t) + minV(s+t)$$

Thus, if we can show that $minV(s+t) \leq maxU(s+t)$ the proof is complete. To do so we start applying 2.11 to A^T, the transpose of A, which gives us

$$min\sum_j A_{.j}y_j \leq max\sum_i A_{i.}x_i \tag{2.17}$$

given that $x_i \geq 0,\ \sum x_i = 1$ and $y_j \geq 0,\ \sum y_j = 1$

We choose x_i and y_j such that

$$U(s+t) = U(0) + (s+t)\sum A_{i.}x_i$$

and

$$V(s+t) = V(0) + (s+t)\sum A_{.j}y_j$$

Now from the proof of Lemma 1 we have that

$$minV(s+t) \leq maxV(0) + (s+t)min\sum A_{.j}y_j$$

combining 2.17 with the definition of a vector system, stating that $minU(0) = maxV(0)$ we get

$$\begin{aligned} minV(s+t) &\leq minU(0) + (s+t)max\sum A_{i.}x_i \\ &\leq maxU(s+t) \end{aligned}$$

QED

We are now ready to complete the proof by a final lemma.

LEMMA 4
For every matrix A and $\varepsilon > 0$ there exists a t_0 such that for any vector system (U, V) it is true that

$$maxV(t) - minU(t) < \varepsilon t, \text{ for } t \geq t_0$$

PROOF:
The proof goes by induction. It is easy to see that it holds for matrices of order 1 since $U(t) = V(t)$, $\forall t$
Now we assume that the theorem holds for all submatrices of A, and then show that it holds for A.
We choose a $\hat{t}$ such that for any vector system (U', V') for the submatrix A' of A we have

$$maxV'(t) - minU'(t) < \frac{1}{2}\varepsilon t \text{ , whenever } t \geq \hat{t}$$

We shall prove that in our given vector system (U, V) for A, if some row or column is not eligible in the interval $(s, s+\hat{t})$ then it is true that

$$maxV(s+\hat{t}) - minU(s+\hat{t}) < maxV(s) - minU(s) + \frac{1}{2}\varepsilon\hat{t} \qquad (2.18)$$

Let us suppose that the k^{th} row is not eligible in the interval $(s, s+\hat{t})$. Then it is possible to construct a vector system (U', V') for the submatrix A', which is equivalent to A with the k^{th} row deleted, in the following manner

$$U'(t) = U(s+t) + C$$
$$V'(t) = Proj_k V(s+t) \text{ for } t = 0, 1, .., \hat{t}$$

In the equations above, C is an n-dimensional vector where

$$C_i = maxV(s) - minU(s) \text{ for } i = 1, 2, .., n$$

$Proj_k V$ is the vector obtained by removing the k^{th} component from V. We shall number the rows of A' as $1, 2, .., k-1, k+1, .., m$.
If (U', V') is a vector system, we know from Definition 5 that $minU'(0) = maxV'(0)$. From the construction procedure we have that

$$\begin{aligned} U'(0) &= U(s+0) + C \\ &= [s_1, .., s_n] + [maxV(s) - minU(s), .., maxV(s) - minU(s)] \\ &= [s_1 + maxV(s) - minU(s), .., s_n + maxV(s) - minU(s)] \end{aligned}$$

Since all the components in $U'(0)$ is summed with the same number, it must be true that the minimum component of $U'(0)$ is the one where $s_i = minU(s)$ and we can therefore see that

$$minU'(0) = minU(s) + maxV(s) - minU(s) = maxV(s)$$

Since $V'(0)$ is a copy of $V(s)$ with the k^{th} component removed, we know that $maxV'(0) = maxV(s)$ since the k^{th} row was not eligible.
Furthermore, for (U', V') to be a vector system, certain recursive restrictions from Definition 5 must be satisfied. It follows from the construction that if

$$U(s+t+1) = U(s+t) + A_{i\cdot} \text{ and } V(s+t+1) = V(s+t) + A_{\cdot j}$$

and we know that k^{th} row is not eligible we have that

$$U'(t+1) = U'(t) + A'_{i\cdot} \text{ and } V'(t+1) = V'(t) + A'_{\cdot j}$$

We can also see from the construction that

$$v_i(s+t) = maxV(s+t) \text{ if and only if } v'_i(t) = maxV'(t)$$

and similarly

$$u_j(s+t) = minU(s+t) \text{ if and only if } u'_j(t) = minU'(t) \text{ for } 0 \leq t \leq \hat{t}$$

Hence we can conclude that U' and V' satisfies the recursive restrictions of a vector system for $0 \leq t \leq \hat{t}$ since U and V do.
From the way we chose $\hat{t}$ we have that

$$maxV'(\hat{t}) - minU'(\hat{t}) < \frac{1}{2}\varepsilon\hat{t}$$

and from the construction of (U', V') we know that it is constructed from $U(s)$ and $V(s)$ and forward, so we can say that

$$maxV(s+\hat{t}) - minU(s+\hat{t}) = maxV'(\hat{t}) - minU'(\hat{t}) + maxV(s) - minU(s)$$

and since

$$maxV'(\hat{t}) - minU'(\hat{t}) < \frac{1}{2}\varepsilon\hat{t}$$

it must be true that

$$maxV(s+\hat{t}) - minU(s+\hat{t}) < maxV(s) - minU(s) + \frac{1}{2}\varepsilon\hat{t}$$

We are now ready to show that given any vector system (U, V) for A it is true that

$$maxV(t) - minU(t) < \varepsilon t \text{ , for } t \geq \frac{8a\hat{t}}{\varepsilon}$$

Consider $t > \hat{t}$, choose $\theta \in [0; 1]$ and $q \in N$ such that $t = (\theta + q)\hat{t}$. We shall divide this proof into two cases.
`Case 1`
Suppose that there exists a positive integer $s \leq q$ such that all rows and columns of A are eligible in the interval $((\theta + s - 1)\hat{t}, (\theta + s)\hat{t})$, and choose the largest such s.
We have a situation as depicted in Figure 2.2

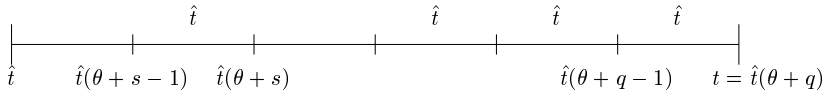


Figure 2.2: *The intervals*

Then we have that in each of the intervals

$$((\theta + r - 1)\hat{t}, (\theta + r)\hat{t}) \text{ , for } r = s + 1, .., q$$

some row or column is not eligible. Thus, by repeated application of 2.18 we get

$$maxV(t) - minU(t) \leq maxV((\theta + s)\hat{t}) - minU((\theta + s)\hat{t}) + \frac{1}{2}\varepsilon(q - s)\hat{t} \tag{2.19}$$

Remember we chose s such that all rows are eligible in the interval $((\theta + s - 1)\hat{t}, (\theta + s)\hat{t})$. From Lemma 3 we get

$$maxV((\theta + s)\hat{t}) - minU((\theta + s)\hat{t}) \leq 4a\hat{t} \tag{2.20}$$

By combining 2.19 and 2.20 we get

$$maxV(t) - minU(t) \leq 4a\hat{t} + \frac{1}{2}\varepsilon(q - s)\hat{t} < (4a + \frac{1}{2}\varepsilon q)\hat{t}$$

`Case 2`
If there exists no such s then we know that in each interval $((\theta+r-1)\hat{t}, (\theta+r)\hat{t})$ we know that some row or column of A is not eligible, and then we have from 2.18 that

$$maxV(t) - minU(t) < maxV(\theta\hat{t}) - minU(\theta\hat{t}) + \frac{1}{2}\varepsilon q\hat{t} \leq 2a\theta\hat{t} + \frac{1}{2}\varepsilon q\hat{t}$$

Therefore we have that in either case

$$maxV(t) - minU(t) < (4a + \frac{1}{2}\varepsilon q)\hat{t} \leq 4a\hat{t} + \frac{1}{2}\varepsilon t < \varepsilon t \text{ , for } t \geq \frac{8a\hat{t}}{\varepsilon}$$

`QED`

Now we are ready to sum up the results from Lemmas 1 to 4.

By combining Lemma 1 with Lemma 4 we get that

$$\lim_{t\to\infty} \frac{maxV(t) - minU(t)}{t} = 0$$

From 2.9 we see that

$$\lim_{t\to\infty} sup\frac{minU(t)}{t} \leq v$$

and

$$\lim_{t\to\infty} inf\frac{maxV(t)}{t} \geq v$$

Hence, we have that

$$\lim_{t\to\infty} \frac{minU(t)}{t} = \lim_{t\to\infty} \frac{maxV(t)}{t} = v$$

which completes the proof of Theorem 4.

Having looked into the core details of the work of Brown and Robinson we are ready to move on. As earlier mentioned, the many of the results outlined in this chapter are based on [Robinson, 1951] which again is based on the unpublished work of George W. Brown. However, we have not seen any of this work applied in practice, which is possibly due to the lack of computer power back in 1949 - 1951 where this work is made. This makes it interesting for us to apply the proposed construction procedure to a few simple games to see how it performs. Furthermore, [Robinson, 1951] suggests an alternative recursive construction procedure and states that it "seems to be" faster in terms of convergence than the one we have given here. This could of course also be interesting to verify. In the next chapter we shall try implementing the suggested procedures.

Chapter 3

Testing Brown's Theorem

Having looked into, and formally proved Brown's theorem, we find it relevant to carry out a few experiments. We intend to test the iterative solution procedure on a few simple games, both symmetric and asymmetric to see if convergence appear.

3.1 jIsol

For the purpose we have developed the program `jIsol` , where *Isol* stands for Iterative Solution, and the j indicates that the program is developed in Java. To use `jIsol`, one needs only to specify the utility matrix, the rest is done by the program. As output one can either get a plot of the bounds, $\frac{minU(t)}{t}$ and $\frac{maxV(t)}{t}$ to see a convergence visualized, or it is possible to get a dump of all the intermediate $U(t)$ and $V(t)$-vectors to see how they change during the procedure, and to see if exact solutions occur.
With these options it is possible to verify both parts of Brown's theorem.

3.2 A Simple Symmetric Game

We choose as a test-bed, the game of scissor-paper-stone which is used to solve many everyday conflicts. Personal experience verifies that it is extremely useful to decide who is to sit on the front seat in the car when going fishing with two pals. However, the original version of the game is designed in a manner such that the best strategy is complete random play. This fact has made us modify the game a bit for this experiment, so more complicated strategies can be beneficial.
What we actually do is to modify the utility matrix such that a victory is not just a victory, but the possible amount of gambling units you win or loose is

P1/P2	Scissor	Paper	Stone
Scissor	0	x	-z
Paper	-x	0	y
Stone	z	-y	0

Table 3.1: A general pay-off matrix

dependent on your choice of hand.
In Table 3.1 below we have included the utility table from Player1's point of view. In the original game of scissor-paper-stone we have that $x = y = z$.
In our version of the game we let $x = 1, y = 2, z = 3$ meaning that if you choose "stone" you have a potential winning of 3 gambling units, but then the potential loss is equally high.
With the redefined utility matrix it seems reasonable to assume that there is a better randomized strategy than $\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\}$. Before we let `jIsol` search for it, we try to find it by a theoretical approach.

3.3 A Theoretical Approach

As mentioned, the task is to solve the game by theoretical considerations. We are assuming that our opponent is intelligent and rational, so pure strategies will lead to loss in the long run. Hence, the task is to find an optimal randomized strategy.

First of all, let us find out what an optimal randomized strategy is. A strategy is optimal if our opponent is indifferent about all of his possible choices, or in other words, the best she can do is to play completely random. Since the game outlined above is symmetric and the utility of a draw is zero for both players, the expected utility in an equilibrium must be zero for all possible choices.
Let us look at the expected utilities from a players point of view

$$\begin{aligned} EU(scissor) &= P(paper)x + P(stone)(-z) \\ EU(paper) &= P(scissor)(-x) + P(stone)y \\ EU(stone) &= P(scissor)z + P(paper)(-y) \end{aligned}$$

Since we just stated that the expected utilities should be zero, we get

$$P(scissor) = \frac{y}{x}P(stone)$$
$$P(paper) = \frac{z}{x}P(stone)$$
$$P(stone) = \frac{x+z}{y}P(scissor) - P(paper)$$

From our definition of the modified version of the game we have that

$$x = 1, y = 2 \text{ and } z = 3$$

Inserting this into the formulas above we get

$$P(scissor) = 2P(stone)$$
$$P(paper) = 3P(stone)$$
$$P(stone) = 2P(scissor) - P(paper)$$

From fundamental probability theory we have that

$$P(scissor) + P(paper) + P(stone) = 1$$

and taking this knowledge into account we get

$$2P(stone) + 3P(stone) + P(stone) = 1$$

And thus,

$$P(stone) = \frac{1}{6}$$

It follows that

$$P(scissor) = \frac{1}{3} \text{ and } P(paper) = \frac{1}{2}$$

Now we know that with the utilities defined in the beginning, the corresponding probability distribution in an optimal strategy is

$$P(hand) = \left\{\frac{1}{3}, \frac{1}{2}, \frac{1}{6}\right\}$$

3.4 The Iterative Solution

Since we have just computed the exact solution we start out searching for the exact solution with `jIsol`. We know that the value of the game is zero for both players so we have a solution of for some t and t' we have that

$$\frac{minU(t)}{t} = 0 = \frac{maxV(t')}{t'}$$

Iteration 1 :	argmaxV(0)	= 2	⇒	U(1)	=	[	-1	0	2]
	argminU(0)	= 2	⇒	V(1)	=	[	1	0	-2]
Iteration 2 :	argmaxV(1)	= 1	⇒	U(2)	=	[	-1	1	-1]
	argminU(1)	= 1	⇒	V(2)	=	[	1	-1	1]
Iteration 3 :	argmaxV(2)	= 1	⇒	U(3)	=	[	-1	2	-4]
	argminU(2)	= 3	⇒	V(3)	=	[	-2	1	1]
Iteration 4 :	argmaxV(3)	= 2	⇒	U(4)	=	[	-2	2	-2]
	argminU(3)	= 3	⇒	V(4)	=	[	-5	3	1]
Iteration 5 :	argmaxV(4)	= 2	⇒	U(5)	=	[	-3	2	0]
	argminU(4)	= 1	⇒	V(5)	=	[	-5	2	4]
Iteration 6 :	argmaxV(5)	= 3	⇒	U(6)	=	[	0	0	0]
	argminU(5)	= 1	⇒	V(6)	=	[	-5	1	7]
Iteration 7 :	argmaxV(6)	= 3	⇒	U(7)	=	[	3	-2	0]
	argminU(6)	= 1	⇒	V(7)	=	[	-5	0	10]
Iteration 8 :	argmaxV(7)	= 3	⇒	U(8)	=	[	6	-4	0]
	argminU(7)	= 2	⇒	V(8)	=	[	-4	0	8]
Iteration 9 :	argmaxV(8)	= 3	⇒	U(9)	=	[	9	-6	0]
	argminU(8)	= 2	⇒	V(9)	=	[	-3	0	6]
Iteration 10 :	argmaxV(9)	= 3	⇒	U(10)	=	[	12	-8	0]
	argminU(9)	= 2	⇒	V(10)	=	[	-2	0	4]
Iteration 11 :	argmaxV(10)	= 3	⇒	U(11)	=	[	15	-10	0]
	argminU(10)	= 2	⇒	V(11)	=	[	-1	0	2]
Iteration 12 :	argmaxV(11)	= 3	⇒	U(12)	=	[	18	-12	0]
	argminU(11)	= 2	⇒	V(12)	=	[	0	0	0]
Iteration 13 :	argmaxV(12)	= 3	⇒	U(13)	=	[	21	-14	0]
	argminU(12)	= 2	⇒	V(13)	=	[	1	0	-2]
Iteration 14 :	argmaxV(13)	= 1	⇒	U(14)	=	[	21	-13	-3]
	argminU(13)	= 2	⇒	V(14)	=	[	2	0	-4]
Iteration 15 :	argmaxV(14)	= 1	⇒	U(15)	=	[	21	-12	-6]
	argminU(14)	= 2	⇒	V(15)	=	[	3	0	-6]

Table 3.2: The Search for the Exact Solution

In Table 3.2 we have included a solution procedure for the simple game described in Section 3.2
We can see from Table 3.2 that we find a solution for $t = 6$ and $t' = 12$. We can also see that we have chosen the first row 2 times, the second row 3 times and the third row 1 time, up until and including the 6^{th} iteration. Hence, we have a solution as follows

$$\left\{\frac{2}{6}, \frac{3}{6}, \frac{1}{6}\right\} = \left\{\frac{1}{3}, \frac{1}{2}, \frac{1}{6}\right\}$$

which is exactly the same we found by our theoretical consideration in Section 3.3.
The same solution can be found by looking at the number of times each column is chosen up until the 12^{th} iteration.
Note that the solution procedure included here is in no way unique, in fact, there is an infinite number of solution procedures since a random choice is made whenever there are more than one v_i and u_j satisfying the recursive restrictions of the definition of a vector system. The solution procedure we have included here is just the one we have found to have the shortest path to an exact solution for both U and V. Various experiments have shown that special cases can occur with more than 1.000 iterations over this same game without an exact solution occurs, and most of the times we need more than 100 iterations before we can verify that $\frac{minU(t)}{t} = \frac{maxV(t')}{t'}$ for some t and t'.
As a final comment on exact solutions, we should mention that there is no guarantee that we will ever find an exact solution but still we can always find an approximate solution as we shall see in the following.

3.4.1 Solution by Convergence

Now let us look at the main result of Brown's theorem stating that if we repeat the iterative procedure again and again we are getting closer and closer to the solution of the game. That is, we can find an approximate solution even if we fail to find an exact one. However, the theorem should still be true if we succeed in finding exact solutions during the recursive process.
To verify this, we repeat the procedure 10.000 times and at each iteration we plot $\frac{minU(t)}{t}$ and $\frac{maxV(t)}{t}$. The result can be seen in Figure 3.1.
From Figure 3.1 we see that both bounds are going against the value zero as we would expect from the theorem. Studying the curves in detail we can see that it looks like both of them are in zero some times and then moving away again. This is of course due to the nature of the solution procedure since there is no opportunity for stopping with an optimal randomized strategy,

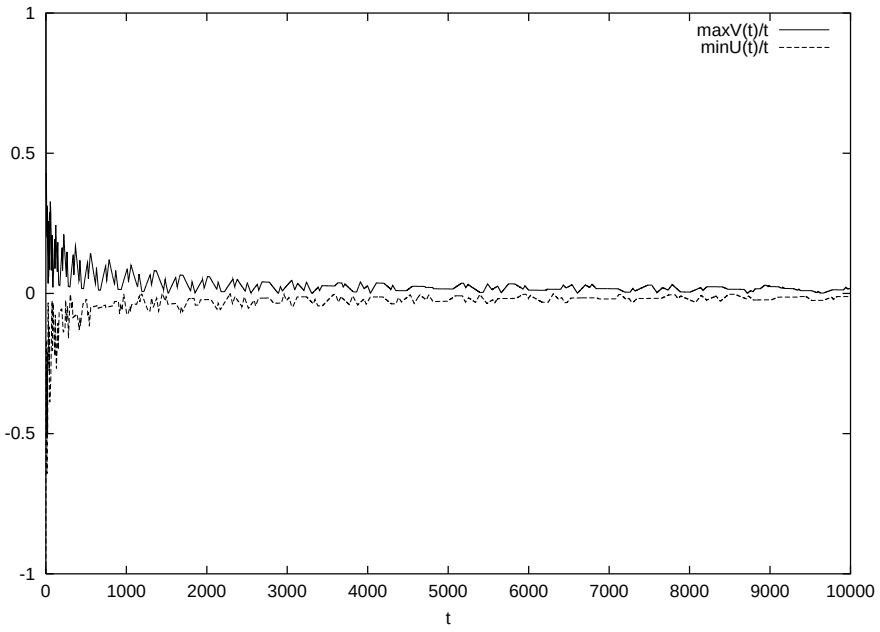


Figure 3.1: *The iterative solution procedure with 10.000 iterations*

not even if it was possible to find such ones at run-time. Again due to the nature of the procedure we also see that as t grows larger the oscillations are getting smaller and smaller.

After 10.000 iterations we read out the following solutions

$$\frac{\text{row count}}{t} = \{0.3358, 0.4933, 0.1709\}$$

and

$$\frac{\text{column count}}{t} = \{0.3371, 0.5036, 0.1594\}$$

The solutions we get are close to the ones we computed and found to be the exact solutions of the game so we can conclude that Brown's theorem is working as expected for symmetric zero-sum games.

As a final experiment to verify Brown's theorem on symmetric games let us try looking into how the two solutions move in order to each other during the recursive solution procedure. We see from the definition that the choices made for the rows are dependent on the current distribution over the columns and vice versa. It therefore seems reasonable to assume that the temporary solutions interact in some manner.

To see the pattern we use the *Euclidean distance* between two probability distributions, defined as

$$dist_E(x, y) = \sum_i (x_i - y_i)^2$$

to see how close they are to each other during the construction.
The result can be seen in Figure 3.2

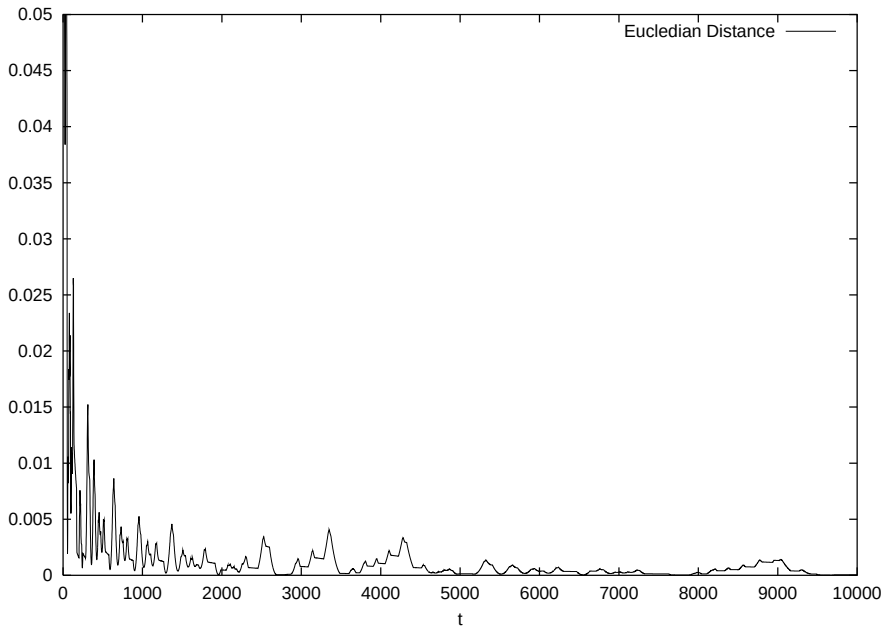


Figure 3.2: *The euclidean distance between the temporary solutions over 10.000 iterations*

Note that the distance is very small all the way through, but in no way constant. It seems that they are moving closer to each other, reach an equilibrium or at least get close to one, and are then forced to move away from each other again. This verifies the conclusion we made when studying how the bounds are moving, stating that even though an equilibrium is reached, the procedure is not stopped. Finally we should note that as t grows larger, the variance in the distance is getting smaller.

3.5 An Asynchronous Selection Procedure

In [Robinson, 1951] it is mentioned that there is another way of constructing vector systems than the one we have described. Remember that the proce-

dure that we are using are based on simultaneous updating of $U(t)$ and $V(t)$. However, it is possible to determine the vectors alternately by replacing the condition on j with the following

$$u_j(t+1) = minU(t+1)$$

The construction procedure is still recursive but when we have formed $U(t+1)$ it is included in the construction of $V(t+1)$ instead of including information on $U(t)$. It is mentioned without further comments that a vector system of this new kind seems to converge more rapidly.
We have tried to verify this statement by plotting the two bounds from the old procedure together with the two bounds from the new procedure to see if faster convergence seems to happen. The result is included in Figure 3.3

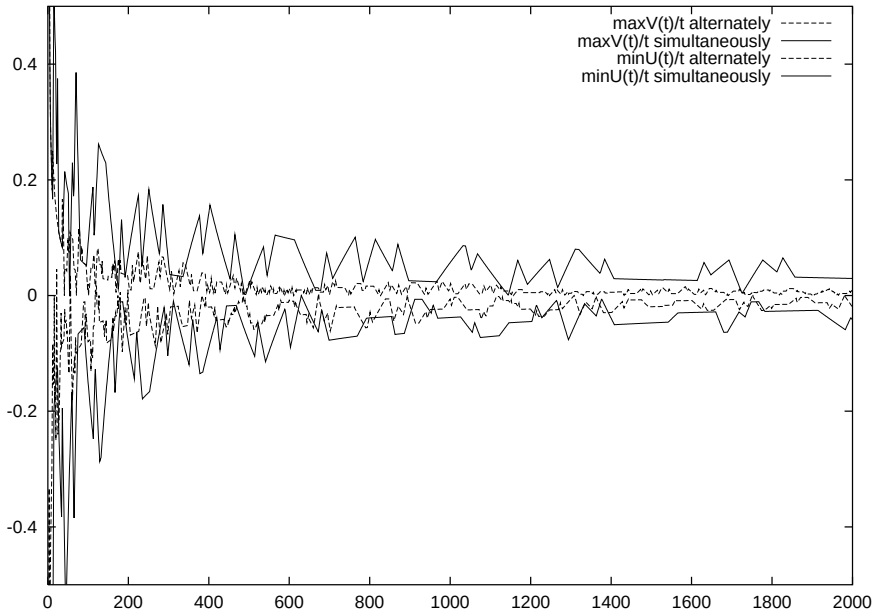


Figure 3.3: *Testing the speed of convergence*

As can be seen it is true that convergence happens faster with the new procedure, which is plotted with dotted lines in Figure 3.3. But this is not the only interesting thing to note. We can also see that the first procedure results in more oscillations where the latter is staying much closer to the value of the game - in this case zero. Therefore it seems like a good idea to use the latter procedure if the task is to get a solution of the game as quickly as possible.

3.6 An Asymmetric Game

Until now we have only tested Brown's theorem on a symmetric zero-sum game or in other words a game with the value zero. Now we intend to modify the game we have used as test bed so far, once again.
This time we let our utility matrix be as follows

$$A = \begin{bmatrix} 1 & 1 & -3 \\ -1 & -2 & 2 \\ 3 & -2 & 1 \end{bmatrix}$$

Note that it is now possible to benefit from a draw. Say that the matrix as we have it here is from Player1's point of view so that if both players choose to play "Paper", he will loose two gambling units, which he of course much pay to Player2.

Since this game is asymmetric it will have a different value for Player1 than for Player2, in fact we have that

$$v_{P1} = -v_{P2}$$

With the symmetric game we knew that the value was zero for both players, and we could therefore easily compute the optimal strategies beforehand. This time we shall do it the other way around - let `jIsol` suggest a solution and see if we can verify it as a set of optimal randomized strategies or in other words, a Nash equilibrium.
Again we see clear tendencies of convergence, apparently centered around the value $-\frac{1}{2}$, indicating that Player1 can expect to loose five gambling units for every ten games. The pattern is clear, but it seems that even as we approach 10.000 iterations we still see large oscillations where both the upper and the lower bound is moving away from what seems to be the value of the game. In other words we could say that apparently the system fails to converge completely.
To prove or disprove this tendency we try to increase the number of iterations to 40.000. The result can be seen in Figure 3.5
As we can see the oscillations are getting smaller, but not much. It seems that we are dealing with a game where the convergence is extremely slow. Since we know of an other construction procedure where we have shown that convergence is not only faster, but also avoiding the oscillations where the bounds are moving away from the value, it could be interesting to see how it performs in this case. The result can be seen in Figure 3.6
As can be seen the oscillations are almost completely gone already after 10.000 iterations with the asynchronous solution procedure, where they were

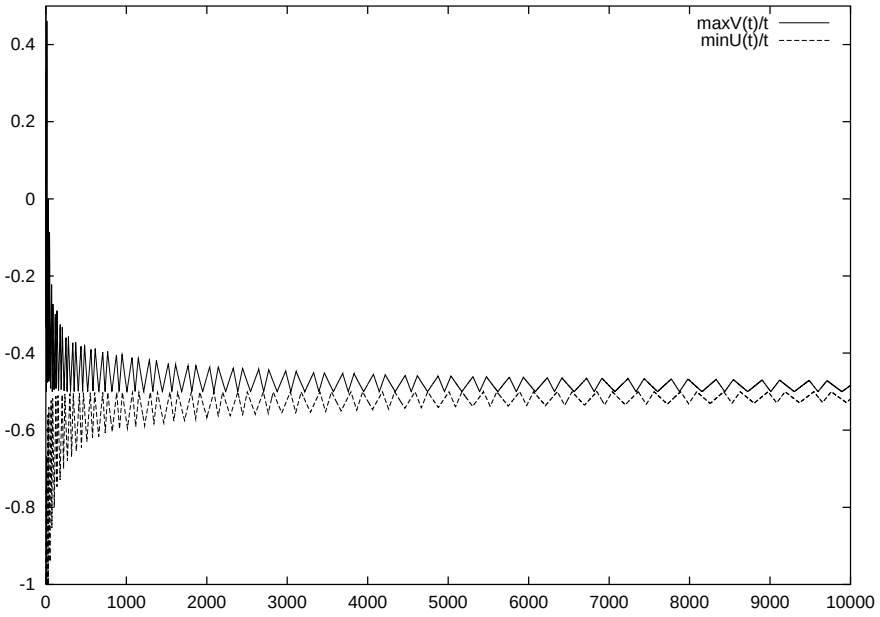


Figure 3.4: *The iterative solution applied to an asymmetric game*

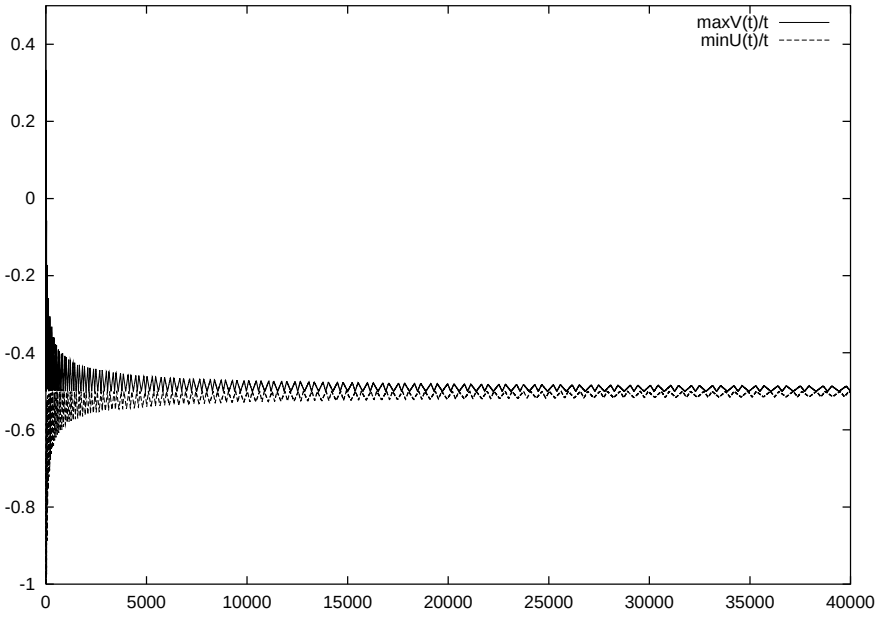


Figure 3.5: *Increased number of iterations*

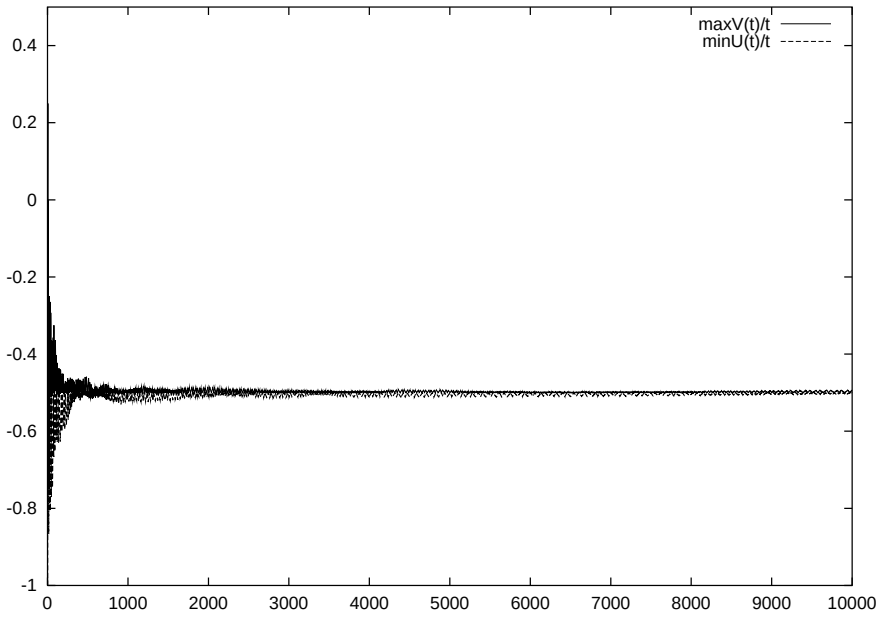


Figure 3.6: *The asynchronous procedure on an asymmetric game*

still significant after 40.000 iterations with simultaneous selection.

3.6.1 The Solution

Now we have said enough about the speed of convergence and it is time to read out the solutions. We get the solutions from the first test - that is, the results are made with simultaneous selection and 10.000 iterations. We get the following

$$\frac{\text{row count}}{t} = \{0.5028, 0.4972, 0\}$$

and

$$\frac{\text{column count}}{t} = \{0, 0.6220, 0.3780\}$$

Let us see if we can verify this result as a Nash equilibrium.
The results we read out are approximate solutions, but it seems that they are converging against

$$\left\{\frac{1}{2}, \frac{1}{2}, 0\right\} \text{ and } \left\{0, \frac{5}{8}, \frac{3}{8}\right\}$$

Let us look at the situation from Player1's point of view. If he knows that Player2 is playing $\{0, \frac{5}{8}, \frac{3}{8}\}$ the situation is

$$EU(scissor) = \frac{5}{8} - 3 \cdot \frac{3}{8} = -\frac{1}{2}$$
$$EU(paper) = -2 \cdot \frac{5}{8} + 2 \cdot \frac{3}{8} = -\frac{1}{2}$$
$$EU(stone) = -2 \cdot \frac{5}{8} + \frac{3}{8} = -\frac{7}{8}$$

Therefore he will never choose to play stone since he will always be worse off by doing so.

From Player2's point of view we have the following situation if we know that Player1 is playing $\{\frac{1}{2}, \frac{1}{2}, 0\}$

$$EU(scissor) = \frac{1}{2} - \frac{1}{2} = 0$$
$$EU(paper) = \frac{1}{2} - 2 \cdot \frac{1}{2} = -\frac{1}{2}$$
$$EU(stone) = -3 \cdot \frac{1}{2} + 2 \cdot \frac{1}{2} = -\frac{1}{2}$$

Thus, since the task of Player2 is to minimize the pay-off to Player1, she should never play scissor. At a first glance it can seem a bit odd that Player2 is preferring paper over stone since the give the same expected pay-off. However there is a reason for this, since the weights between them as they are in this solution is solving the task of letting the best strategy of Player1 be randomized play. In other words, if Player2 played a different randomized strategy over paper and stone, Player1 could benefit from changing his strategy. Thus, the strategies found are a Nash equilibrium of the game.

3.7 Solving a Game During Play

We can now conclude that Brown's theorem works in practice for solving a game. However, for several reasons, the solution procedure is only suitable for solution of a game before the game begins, and not for finding an optimal solution during the play against an opponent. First of all, if we were to use this procedure to find run-time solutions of games, we would not be able to use the asynchronous selection procedure since this would mean that we

would have to ask our opponent to tell us what decision she made before we make our own, but since she is assumed to be both intelligent and rational she would probably find that to be a bad idea.

Secondly, Brown's iterative solution procedure is based on selection from a maximum criterion or in other words, select what seems best and nothing else. However, we have from [Myerson, 1991] that in order to reach optimal play, one must follow an optimal randomized strategy, and make a weighted selection over the expected utilities to avoid that a *counting opponent* will know your deterministic strategy. The term *counting opponent* might need a bit explanation. If we during a game always made the decision giving us the maximum expected pay-off, an intelligent opponent would be able to keep track of what decision is giving us the maximum expected pay-off at any time and therefore use this knowledge in his decision.

Brown probably never intended his method to be suitable for implementing what today is known as *intelligent agents*, but it would surely be interesting if we could use the idea behind the iterative solution procedure to implement such an agent. Brown's theorem is only designed to solve games with one decision so in the following chapter we shall try implementing intelligent agents for one-decision two-player zero-sum games.

Chapter 4

Learning One-Decision Bayesian Networks

Having verified Brown's theorem both in theory and practice, it is time to see if we can apply the ideas in other areas of decision theory. Especially we are interested in implementing intelligent agents with the ability to find optimal strategies for any finite zero-sum game they are set to play. Since Brown's procedure provides us with the ability to solve a game it is natural to see if we can integrate it into a scheme upon which we can implement intelligent agents.
One of the most promising technologies of today when talking decision support systems is *Bayesian networks* as defined in [Jensen, 1996], so our main task shall be to find out if we can integrate Brown's solution procedure into Bayesian networks.

4.1 Training Scheme

To introduce the iterative solution procedure into Bayesian networks we need a training scheme corresponding to Brown's method of counting cases.
From [Jørgensen, 2000] we get the definition of the training scheme called *fractional updating*, also used and extended with the concept of *fading* in [Olesen et al., 1992].
To ensure a clear understanding of fractional updating let us look at a simple example where we apply the ideas. Say we have three variables A, B and C each with three states, where B and C are parents of A. We assume local- as well as global independence in this network and we can therefore consider

$$P(A|b_i, c_j) = (x_1, x_2, x_3)$$

as a distribution we have reached by observing several cases where (B, C) were in the state (b_i, c_j).
Now we have to express our certainty of this distribution by what is called a *sample size*.
We include the sample size, s in a table

$$n = (n_1, n_2, n_3) = (sx_1, sx_2, sx_3)$$

where $n_1 + n_2 + n_3 = s$
Thus, we can say that n_i is the number of times we have seen A in state i, eg if we choose $s = 30$ and have that $x_1 = x_2 = x_3 = \frac{1}{3}$ we can say that we have observed A in each state ten times. As can be seen, the larger sample size, the larger certainty of the initial distribution.
Now when we see a new case, say the case where A is in state 2, and (B, C) is in state (b_i, c_j) we count up s and n_2, yielding the new distribution

$$(x_1^+, x_2^+, x_3^+) = \left(\frac{n_1}{s+1}, \frac{n_2+1}{s+1}, \frac{n_3}{s+1} \right)$$

As mentioned, [Olesen et al., 1992] introduced the concept of *fading* in Bayesian networks, which we also used in [Jørgensen, 2000]. The purpose of *fading* is to make the networks "forget" what they have learned in the past so they can easily adapt to a new context if this changes. To do so, a *fading factor* $q \in [0 : 1]$ is introduced. This value is multiplied onto the sample size to keep it from growing into extreme values.
In practice this means that when we run a case the new sample size is $qs+1$, and running n cases yields a sample size of

$$q^n s + \frac{1 - q^n}{1 - q}$$

Note that

$$\lim_{n \to \infty} \left(q^n s + \frac{1 - q^n}{1 - q} \right) = \frac{1}{(1 - q)}$$

This means that when running several cases so n grows large, the effective sample size can be computed as $1/(1-q)$. So if $q = 0.95$ we have an effective sample size of 20.

4.2 The `MYELIN` tool-box

In [Jørgensen, 2000] we developed a general tool-box, `MYELIN`, for working with adaptive Bayesian networks. All tests concerning Bayesian networks in

this thesis is created using a new version of `MYELIN` which is developed in Java to work with the newly released HUGIN Java API. The new version of `MYELIN` contains all of the old methods for performing probability updating with and without fading and making decisions based on the modified probabilities. Furthermore we have included methods for computing distances between probability distributions, dumping expected utilities at any time and various tools to simulate dices and coins.

Decision making in `MYELIN` can be performed in different ways, so we can always do what is most suitable for the tests we need to perform. That is, we have included methods for playing only on the maximum expected utility, to be used in search of solutions, as well as we have methods making decisions over all the expected utilities to avoid deterministic play.
Whether or not to use fading can be determined per experiment. Since fading is a concept developed for adapting into changing contexts, we shall not use it in the tests made for verifying Brown's solution procedure in Bayesian networks. However, we intend to carry out a single experiment to see if the idea behind Brown's theorem still holds, extended with the concept of fading, making it even more suitable for adaptive behavior in games.

4.3 Experiments

As earlier mentioned the main purpose of this chapter is to find out if it is possible to integrate Brown's solution concept into Bayesian networks. We know that Brown's theorem corresponds to counting cases as is also the case in fractional updating. In other words the task is to verify that intelligent agents bases on adaptive Bayesian networks using fractional updating are able to find a solution of the game they are set to play.

We have decided to use the same simple game as we did in the previous chapter, namely the game of scissor-paper-stone with various modified utility functions.

4.3.1 The Set-Up

In Figure 4.1 the Bayesian network used for this test is shown. As can be seen it is very simple, the only things to say about it is that the node labelled "Utility" reflects the utility matrix which we will vary a few times during the tests. In the initial probability distribution for the node "Opponent", the probability is $\frac{1}{3}$ for both scissor, paper and stone.

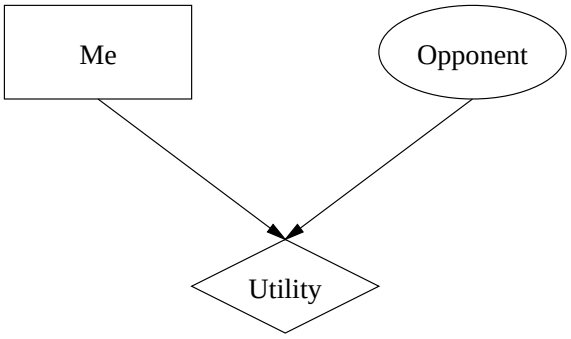


Figure 4.1: *The Bayesian network used for Scissor-Paper-Stone*

Utilities

In the first experiment we want to test a symmetric version of the game, so we use the same utility matrix as in Section 3.4.
For convenience we have included the utility matrix here

$$A = \begin{bmatrix} 0 & 1 & -3 \\ -1 & 0 & 2 \\ 3 & -2 & 0 \end{bmatrix}$$

With these utilities we have already computed a Nash equilibrium in Section 3.3 and verified it in practice in Section 3.4 so we expect the outcome of this experiment to be a situation where we have our "Opponent" distribution to be

$$P(Opponent) = \left\{ \frac{1}{3}, \frac{1}{2}, \frac{1}{6} \right\}$$

for both players.

For training purposes we use fractional updating to update the probability distribution of the node "Opponent". Both players are allowed to adapt at the same time, so the interesting question is whether they will converge against the same final probability distribution when the game is symmetric, and if they do so, is this distribution a Nash equilibrium ?

For decision making we follow Brown's idea and let the players choose only the decision with maximum expected utility. From [Myerson, 1991] we know that this is not the optimal way of playing since it is possible for our opponent to predict our strategies at any time by keeping track of the same data as we do. However, we showed in [Jørgensen, 2000] that is does not make

significant influence on the final distributions if we play only on the one with the maximum expected utility or if we use the expected utilities to weigh the possible choices.
We let the players face each other 50.000 times which should be more than sufficient for a convergence to appear. The outcome of the games is that Player1 has lost 94 gambling units to Player2. Since a difference of 94 gambling units out of 50.000 games is sufficient close to zero, this verifies the fact that the game in this test was symmetric and the value therefore is zero for both players.

4.3.2 Results

Now let us look at the final probability distributions for the players. Player1 ends up with the following

```
Opponent = {0.3304, 0.5038, 0.1659}
```

Similarly we include the final distribution for Player2

```
Opponent = {0.3295, 0.5069, 0.1636}
```

As can be seen, the players end up with distributions that are very much alike. The Euclidean distance between the results is calculated to be

$$dist_E(P1, P2) = 1.5912 \times 10^{-5}$$

This verifies that we can use adaptive Bayesian networks with the training scheme of fractional updating to implement adaptive agents for zero-sum games with one decision since both agents converged against the pre-computed Nash equilibrium.
To see how this looks from a players point of view we try to dump the expected utilities for Player1 at the end of the series of games. These are shown below

```
Me = {0.0062, 0.0014, -0.0165}
```

The interesting thing is that they are close to zero - the value of the game - for all possible choices, meaning that when the opponent is using the strategy shown above, the player facing him is indifferent about what choice to make. In other words, it is not more beneficial to choose one over another.

4.3.3 Another Utility Function

To verify this interesting tendency we try to change the utility function and repeat the experiment once again. The utility matrix used for this second experiment can be seen below.

$$A = \begin{bmatrix} 0 & 3 & -3 \\ -3 & 0 & 2 \\ 3 & -2 & 0 \end{bmatrix}$$

As before we repeat the game 50.000 times and the results turn out as shown below.
From Player1's point of view the final distribution is as follows

```
Opponent = {0.2513, 0.3750, 0.3737}
```

And from Player2 we get

```
Opponent = {0.2527, 0.3737, 0.3736}
```

By using the formulas from Section 3.3 we find the solution to be

$$P(hand) = \left\{ \frac{1}{4}, \frac{3}{8}, \frac{3}{8} \right\}$$

So again we get a confirmation that adaptive Bayesian networks can solve the same problems as Brown's procedure, and even solve them while playing. As in Brown's solution we can also read out an approximate value of the game by calculating the average winnings for a player.
The Euclidean distance between the two players probability distributions after this experiment is

$$dist_E(P1, P2) = 3.7084 \times 10^{-6}$$

which smaller than in the first experiment. However, observations during the games show us that the distance is varying all the time, so this does not say anything like "There is even more convergence in this version of the game, since the final distance is smaller". More likely we should conclude that the game ended at a moment where the distance was small in the latter experiment.

Oscillations in Distance

To see how the distance between the probability distributions is varying during the experiment, we try starting the players out with two completely different distributions to get a high distance in the beginning. Player1 gets the following

```
Opponent = {1, 0, 0}
```

while Player2 is started out with

```
Opponent = {0, 0, 1}
```

We repeat the game 10.000 times and get the variation pattern included in Figure 4.2

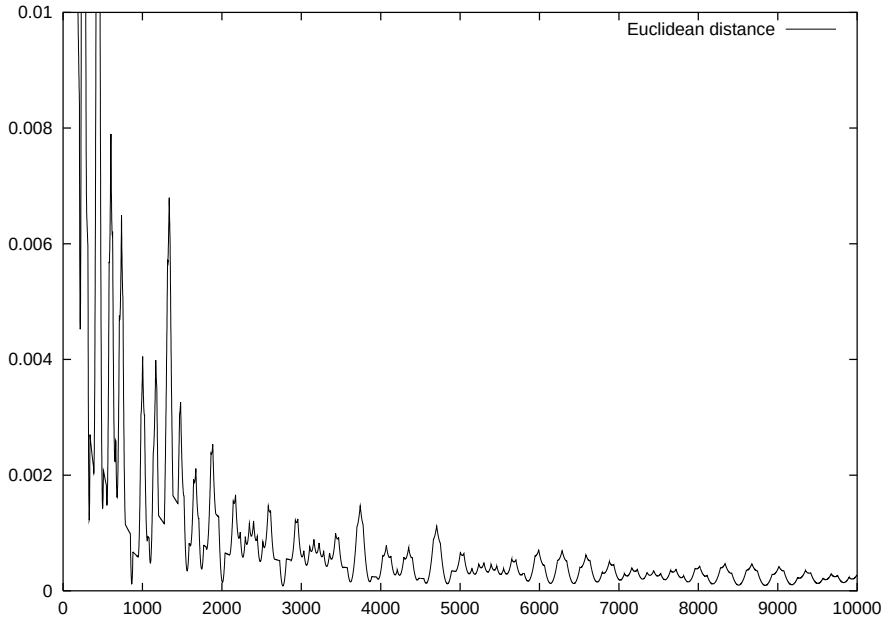


Figure 4.2: *The Euclidean distance during the games*

As can be seen we get a pattern similar to the one we saw in Section 3.4 where the players are moving in order to each other all the time which results in some tiny oscillations around a distance of zero.

4.3.4 Asymmetric Games in Bayesian Networks

We ended our experiments with Brown's solution procedure with a verification of it working on asymmetric games as well as symmetric. We shall in the following see if we can use adaptive Bayesian networks for solution of asymmetric games. This is done to see if the implemented agents are capable of finding their own optimal strategies when they are different from those of their opponent.
We use the same utility function as we used to verify Brown's procedure, meaning that our utility matrix is as follows for Player1

$$A_{P1} = \begin{bmatrix} 1 & 1 & -3 \\ -1 & -2 & 2 \\ 3 & -2 & 1 \end{bmatrix}$$

and for Player2 we have

$$A_{P2} = \begin{bmatrix} -1 & 1 & -3 \\ -1 & 2 & 2 \\ 3 & -2 & -1 \end{bmatrix}$$

The reason for including two utility matrices in this experiment is that it is not possible to represent asymmetric games in a single Bayesian network as it is with symmetric games. Thus we have two versions of the Bayesian network in Figure 4.1 but they only differ on the utilities.
We repeat the game 10.000 times which have shown in the other test to be more than sufficient for a convergence to appear
Player1 ends up with the following distribution

```
Opponent = {0.0007, 0.6252, 0.3741}
```

and Player2 ends with

```
Opponent = {0.5044, 0.4931, 0.0026}
```

Note the Player1's distribution is reflecting the behavior of Player2. Therefore the solution we can read out here is that the randomized strategy of Player1 is approximately

$$P(hand)_{P1} = \left\{ \frac{1}{2}, \frac{1}{2}, 0 \right\}$$

and for Player2 we have

$$P(hand)_{P2} = \left\{0, \frac{5}{8}, \frac{3}{8}\right\}$$

which we showed in Section 3.6 is a Nash equilibrium.
In an asymmetric game we can of course not expect the distance between the two distributions to be zero since the players must use different strategies, but we can still expect the distance to be anchored around the distance between the two exact solutions. We compute this distance to be

$$dist_E(P1_{exact}, P2_{exact}) = \sum_i (x_i - y_i)^2_{exact} = \frac{1}{2}^2 + \frac{1}{8}^2 + \frac{3}{8}^2 = \frac{13}{32}$$

In Figure 4.3 we have plotted the distance varying over the games together with the value we just computed. This experiment is carried out to verify that even if the game is asymmetric, the behavioral pattern for the players is the same.
We see from the figure that the distance is actually oscillating around the pre-computed value and the pattern is the same as in Section 3.6 where the oscillations almost fails to fade out. However, Brown's theorem is not mentioning anything about the speed of convergence so this is not a problem.

4.3.5 Fading

As a final experiment with the integration of Brown's method into one-decision Bayesian networks we try extending the training scheme with fading. This is done to verify that intelligent agents using fading are still capable of solving the games they are set to play, even though they are using fading. The reason that we are interested in such an experiment is that if we can verify this, we have agents that can solve a game as suggested by Brown, but furthermore they are able to adapt to a new behavior if the opponent is changing his strategy.

Again we use the symmetric version of the game. We set the fading factor to be $q = 0.99$ yielding an effective sample size of 100. The game is repeated 10.000 times and the final distributions are as follows

Player1 has found Player2 to be playing

```
Opponent = {0.3246, 0.4915, 0.1839}
```

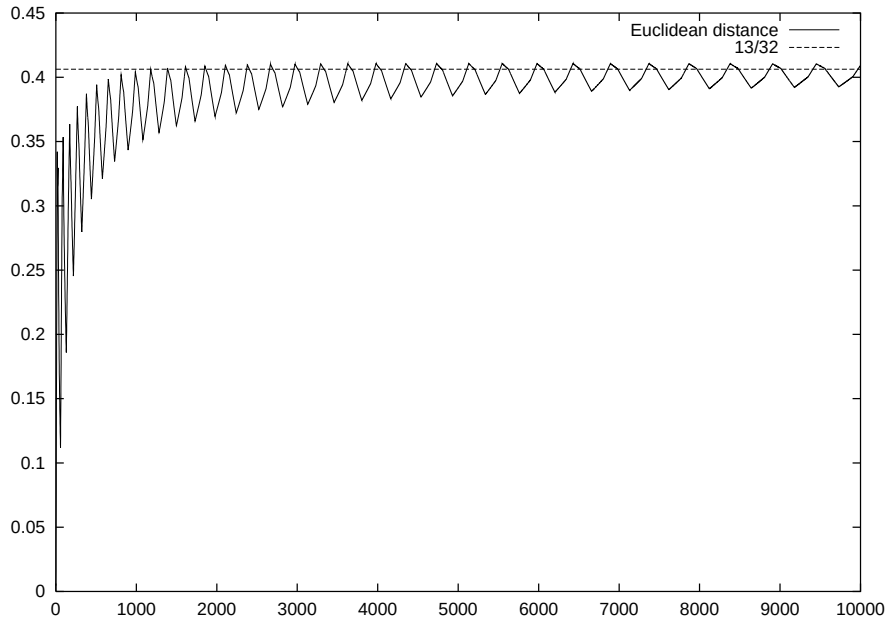


Figure 4.3: *The Euclidean distance during the games*

and Player2 has found Player1 to be playing

$$\texttt{Opponent} = \{0.3247,\ 0.5120,\ 0.1634\}$$

So we get a confirmation of Brown's solution procedure integrated in Bayesian networks is still valid if we use the concept of fading.

4.3.6 Summary

We have now shown that adaptive Bayesian networks can be used for solution of finite two-person zero-sum games with one decision. This is interesting in the area of intelligent agents since you can place an agent based on this technology in any two-person zero-sum game and he will be able to find the optimal randomized strategy for this game.
We mentioned earlier that this optimal strategy is optimal only when the opponent is intelligent and rational. Thus, if we face an opponent playing the same static strategy in all the games we could benefit from playing a strategy that is maximizing our pay-off against this special opponent. Brown's solution procedure is naturally unable to exploit potential weaknesses of opponents, but we have shown in [Jørgensen, 2000] that when using fractional

updating for adaptive Bayesian networks we get agents that are able to adapt to the strategy that is optimal against any opponent they are facing. Furthermore, we have verified that we can extend Brown's solution procedure with the concept of fading and it is still valid. Thus, we get intelligent agents that can adapt to a changing strategy of the opponent, and thereby also exploit potential weaknesses.

It seems so far that Bayesian networks as foundation for intelligent agents for two-person zero-sum games is a very good set-up. If we could apply these ideas to more complex games we would have a very strong representation of adaptive intelligent agents. In the following chapter we shall try doing so.

Chapter 5

Learning Two-Decision Bayesian Networks

The main task of this chapter is to find out if we can apply the ideas from the previous chapters to more complex games. Brown's theorem is only valid for games with one decision but having applied the idea into adaptive Bayesian networks it seems reasonable to assume that we can use it for more complex games, for example a game with more than one decision.
To verify this assumption we have designed a game, *two-person high/low* for the purpose.

5.1 The Rules for Two-Person High/Low

- Both players have two "3-sided" dices with numbers 1 to 3
- Both players pay 1 gambling unit to participate
- The game starts with both players throwing both dices without showing the result to the opponent
- After having viewed the result, both players must choose a dice which they will show to the opponent
- Finally the players must make their bids. Each player has to guess if the sum of his dices is higher or lower than the sum of his opponents
- If one player is playing "Low" and the other is playing "High" the game is a draw
- If the sums are equal the game is a draw

- The winner is the one with the correct bid
- The winner takes the pot

As can be seen the game has lots of possibilities for ending with a draw, and therefore it is probably not a game suitable for settling who is to buy the next round of beer or so. However, this does not matter in our case, since the game is designed specifically to be suitable for verifying simple game theoretic concepts in a more complex set-up. Even though the game seems simple at a first glance, it is actually pretty complex in theoretical terms since the game includes both more than one decision per player and private information - actually the players can choose what part of their information they want to keep private. So after all the game seems complex enough to fulfill its task, namely being a complex test bed for adaptive behavior in games.
Unfortunately, the game being so complex makes it very difficult to pre-compute a Nash equilibrium of the game, so we will instead have to see if we can verify the results as being a Nash equilibrium by arguing that the strategies reflect intelligent and rational behavior.

5.2 Bayesian Model

With the basic rules outlined in the previous section we see that the game is symmetric, and since both players are to make their bids simultaneously we need only a single Bayesian network which both players can share. Of course they get their own private instant of the network in which they can perform probability updating.

The network we use can be seen in Figure 5.1
A few notes about the network design might be needed.
The nodes MyDice1, MyDice2, OppDice1 and OppDice2 are used to enter the value of the dices we get from `MYELIN`. MyHand and OppHand are used to transform the two dices into a *hand type* which can be one of the following:

"1-1", "1-2", "1-3", "2-2", "2-3" or "3-3"

The reason for performing this translation from the two dices into a hand type is to save states in the table where we perform probability updating since there is no reason to make a distinction between the case where Dice1 is "1" and Dice2 is "2" and the case where Dice1 is "2" and Dice2 is "1". The leftmost utility node is prohibiting a player from showing a dice he does not have, and the decision nodes Show, OppShow, MyBid and OppBid should be self explaining.

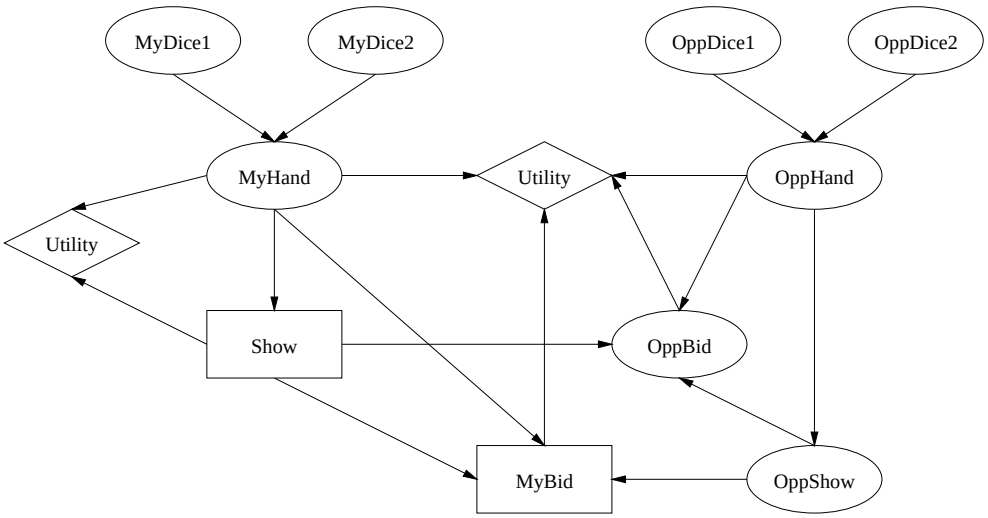


Figure 5.1: *The Bayesian network used for two-person high/low*

The utility node in the middle is used to represent the utility function as defined in the rules of the game.

5.3 Experiments and Results

Having designed a Bayesian network representing the game, we are now ready to implement two intelligent agents based on this network. As usual we use `MYELIN` in order to perform the probability updating, computing distances and printing the results.

The adaptive nodes are OppShow and OppBid which we update according to the observations during the game. We have decided to use perfect hindsight meaning that both players have to show their hidden dice after each game.

With the simpler games we found that 10.000 iterations of the game was more than sufficient for a convergence to appear. However, since we can expect a slower convergence now when we have two adaptive nodes at the same time, we raise the number of experiments to 50.000.

As a final note before looking at the results we mention that we actually use a selection procedure a little different from the one Brown suggested for games with one decision. Instead of always selecting the node with the maximum expected utility, we use a weighted selection procedure. Why this is done will be discussed in the end of this chapter where we have introduced the problems forcing us to use this new procedure.

5.3.1 Final Potentials

After the 50.000 games we first note that Player1 has won 4953 gambling units and Player2 has won 4872. Since they have won them from each other we can also say that Player1 has won 81 gambling units from Player2. Since 81 out of 50.000 is sufficiently close to zero, we take this as a confirmation of the game being symmetric. Furthermore we should note that it is only approximately 20% of the games where a winner is found, the rest of the games are draws. This is also expected since the game is designed in a way such that draw games easily occur.

Now let us look at the final potentials for each of the players. Player1's potential over which dice Player2 is showing given a hand type is shown in Figure 5.2 and the same potential from Player2 is shown in Figure 5.3.
A short note on how to read the figures might be needed. We have the parent state in the rightmost column telling us which hand type our opponent had, and the three data columns in the distribution tells us which dice she will tend to show in this parent situation. The states are "1", "2" and "3".

```
potential (OppShow | OppHand)
{
 data = (( 1 0 0 )                  %  1-1
         ( 0.000792098 0.999208 0 ) %  1-2
         ( 0.50015 0 0.49985 )      %  1-3
         ( 0 1 0 )                  %  2-2
         ( 0 0.999248 0.000751614 ) %  2-3
         ( 0 0 1 ));                %  3-3
}
```

Figure 5.2: *Player1's distribution over Player2's choice of dice to show*

```
potential (OppShow | OppHand)
{
 data = (( 1 0 0 )                  %  1-1
         ( 0.00105013 0.99895 0 )   %  1-2
         ( 0.499776 0 0.500224 )    %  1-3
         ( 0 1 0 )                  %  2-2
         ( 0 0.999068 0.000931842 ) %  2-3
         ( 0 0 1 ));                %  3-3
}
```

Figure 5.3: *Player2's distribution over Player1's choice of dice to show*

First of all we see that the two potentials show the same overall pattern. We shall look further into the distance between the final distributions later on, but for now we conclude that they are alike.
We have three trivial hand types, "1-1", "2-2" and "3-3" where there is no actual choice of which dice to show, as can be seen these are updated correct. Next we can see that if one of the agents have a hand with a dice showing "2", this dice is shown to the opponent. This is a rational behavior since the opponent will have no clue whether the hand is "High" or "Low", since it can be either of type "1-2", "2-2" or "2-3".
The last possible hand is "1-3" and we can see that the agents are playing a randomized strategy, $\{\frac{1}{2}, \frac{1}{2}\}$. This is also what we could expect since the utility function is symmetric and there is no benefit of trying to win on a high hand compared to try winning on a low. Therefore there is no reason to prefer showing "1" over "3" or the other way around.

Now let us try looking at the distribution of the other adaptive node, namely OppBid. It is a bit more complicated to read data out from this one so an example might be needed

```
potential (OppBid | Show OppHand OppShow)
{
 data = (((( 0.00224215 0.997758 )      %  1  1-1  1
          (( 0.5 0.5 )                  %  1  1-2  1
...
```

Figure 5.4: *Example*

If we look at Figure 5.4 it must be read in the following way: The first data line tells us that in the situation where we have showed our opponent a "1", she has got a hand of type "1-1" and she has shown us "1" she will most likely bid on "Low". That is, the first number is the probability that she is playing "High" and the second number is the probability that she is playing "Low".

The final distributions over OppBid from the two players are included in Figures 5.5 and 5.6.
Note that we have trimmed all the impossible configurations away from the figures, eg the ones where the opponent is showing a dice she does not have, like for example the situation "1 1-1 2". Due to the nature of the Bayesian network these are represented when running the test, but since they are never chosen and therefore not counted up, we have removed them from the figures to save some space.

```
potential (OppBid | Show OppHand OppShow)
{
 data = ((((  0.00224215 0.997758 )      %  1  1-1  1
           (( 0.5 0.5 )                  %  1  1-2  1
            ( 0.502936 0.497064 )        %  1  1-2  2
           (( 0.997554 0.00244618 )      %  1  1-3  1
            ( 0.998013 0.00198728 ))     %  1  1-3  3
            ( 0.99774 0.0022604 )        %  1  2-2  2
            ( 0.998901 0.0010989 )       %  1  2-3  2
            ( 0.583333 0.416667 ))       %  1  2-3  3
            ( 0.997852 0.00214777 )))    %  1  3-3  3
          ((( 0.000832501 0.999167 )     %  2  1-1  1
           (( 0.5 0.5 )                  %  2  1-2  1
            ( 0.000412337 0.999588 )     %  2  1-2  2
           (( 0.484292 0.515708 )        %  2  1-3  1
            ( 0.508264 0.491736 ))       %  2  1-3  3
            ( 0.482741 0.517259 )        %  2  2-2  2
            ( 0.999593 0.000406901 )     %  2  2-3  2
            ( 0.6875 0.3125 ))           %  2  2-3  3
            ( 0.999209 0.000791139 )))   %  2  3-3  3
          ((( 0.00186986 0.99813 )       %  3  1-1  1
           (( 0.416667 0.583333 )        %  3  1-2  1
            ( 0.000942685 0.999057 )     %  3  1-2  2
           (( 0.00196232 0.998038 )      %  3  1-3  1
            ( 0.00157928 0.998421 ))     %  3  1-3  3
            ( 0.00184094 0.998159 )      %  3  2-2  2
            ( 0.497269 0.502731 )        %  3  2-3  2
            ( 0.611111 0.388889 ))       %  3  2-3  3
            ( 0.998285 0.00171468 ))));  %  3  3-3  3
}
```

Figure 5.5: *Player1's view of Player2*

```
potential (OppBid | Show OppHand OppShow)
{
 data = ((((  0.00222618 0.997774 )      %  1  1-1  1
           ((  0.5 0.5 )                 %  1  1-2  1
            (  0.496226 0.503774 )       %  1  1-2  2
           ((  0.99753 0.00247036 )      %  1  1-3  1
            (  0.998192 0.00180766 ))    %  1  1-3  3
            (  0.997736 0.00226449 )     %  1  2-2  2
            (  0.99891 0.0010898 )       %  1  2-3  2
            (  0.5 0.5 ))                %  1  2-3  3
            (  0.997966 0.00203417 )))   %  1  3-3  3
          (((  0.000807754 0.999192 )    %  2  1-1  1
           ((  0.5 0.5 )                 %  2  1-2  1
            (  0.000421017 0.999579 )    %  2  1-2  2
           ((  0.490782 0.509218 )       %  2  1-3  1
            (  0.492478 0.507522 ))      %  2  1-3  3
            (  0.500167 0.499833 )       %  2  2-2  2
            (  0.999603 0.00039733 )     %  2  2-3  2
            (  0.5 0.5 ))                %  2  2-3  3
            (  0.999192 0.000807754 )))  %  2  3-3  3
          (((  0.00182749 0.998173 )     %  3  1-1  1
           ((  0.416667 0.583333 )       %  3  1-2  1
            (  0.000938086 0.999062 )    %  3  1-2  2
           ((  0.00237192 0.997628 )     %  3  1-3  1
            (  0.00150602 0.998494 ))    %  3  1-3  3
            (  0.00185874 0.998141 )     %  3  2-2  2
            (  0.502625 0.497375 )       %  3  2-3  2
            (  0.583333 0.416667 ))      %  3  2-3  3
            (  0.998194 0.00180636 ))));  %  3  3-3  3
}
```

Figure 5.6: *Player2's view of Player1*

By looking at the figures we see that they share the same overall pattern or in other words, they are converging against the same randomized strategies. Again, we shall return to considerations about the distance later on.
In the following we shall see if we can verify the strategies in Figures 5.5 and 5.6 as a Nash equilibrium.

5.3.2 Pollution

Notice that in some situations the agents seems to have been playing by different randomized strategies, eg a situation like "2 2-3 3". Does this indicate that they have converged against different randomized strategies or can we find a better explanation ?

We shall refer to this phenomenon as *pollution*. It occurs due to the initial distribution where the probabilities for all possible choices are equal. Therefore it can happen that if you have a hand of type "2-3" you choose to show the "3" since you have not yet discovered that it is beneficial to always show the "2". Of course your opponent will take these cases into account, and count up his probability distributions according to what he is observing. Unfortunately, it turns out that all strategies where you have a "2" and does not show it are dominated and therefore these configurations are never played again so the probabilities remain unchanged. For example in the situation "2 2-3 x", Player1 is only showing "3" three times during the 50.000 games, in contradiction to showing "2" 6144 times.

We conclude that we can not expect configurations based on dominated strategies in a parent node to have a reasonable distribution, and we refer to this phenomenon as pollution.

5.3.3 The Strategies

As a first step in verifying the final distributions as a Nash equilibrium we shall try looking at some of the configurations and see if we can explain the suggested strategies as reflecting intelligent and rational behavior.
The first configuration we look at is also the first in Figure 5.5.
We see that in the situations "x 1-1 1" both players are consequently playing "Low". Is this rational ?

If your hand is "1-1" you can never loose by playing "Low" and you can never win by playing "High" so it seems reasonable that you would only want to play "Low". Furthermore, if your opponent realizes that you are playing

deterministically in this configuration she cannot use this to win since she can never have a lower hand. One could argue that she could just play "High" to ensure a draw, but she have no chance of knowing if the "1" we showed her indicates that we have a hand of type "1-1" and "1-3" so there is a risk involved in playing "High" for some hand types, and still it can never lead to a winning.

Reverse arguments can of course be used for situations of the kind "x 3-3 3".

The most interesting situations occur when a "2" is showed since this introduces the most uncertainty for the opponent.

We try looking at the situation "2 1-3 1" which indicates that we have showed "2" to our opponent, she has a hand of type "1-3" and she has shown the "1" to us. From the figures we see that she is indifferent between playing "High" and "Low" which is also the case if she has shown her "3" to us. Is this rational ?

Since we showed her a "2" she does not know whether we have "1-2", "2-2" or "2-3", so all she can conclude is that the sum of our hand is either one smaller than, the same as, or one larger than her hand. Furthermore, the probabilities for our hand being smaller or larger are the same, and it is less probable that we have a hand of type "2-2". The expected utility of us having the same sum is zero, so this difference in probability is removed, and we can therefore conclude that it is rational to be indifferent between "High" and "Low".

The same kinds of arguments as we have seen here can be used to explain the rationality of all the remaining unpolluted configurations, so still it seems that the strategies from the final potentials are satisfying what it takes to be optimal strategies. As a further verification we shall try looking into the expected utilities a player has when he has observed these final strategies as being the behavior of his opponent.

5.3.4 Expected Utilities

In this subsection we shall look into what the final potentials means for the expected utilities for the players. We know from earlier chapters that if we have an equilibrium the players must be indifferent about their possible choices. However, in this game the players have some private information, so a player might know that he can never win by playing on "Low" in a given situation, and this will of course influence on his expected utilities.

Show

First we try looking at what dice we should show in any situation. As earlier mentioned the cases "1-1", "2-2" and "3-3" are trivial in terms of which dice to show, so we shall not consider them here. We shall consider ourselves as Player1 and we shall discuss the situations from our own point of view.

In the situation where we have a hand of type "1-2" our expected utilities are as follows

```
1-2 : Show = {-0.1095, 0.0586, -4.6663}
```

We see that we should show "2" which we argued in the previous section is a rational behavior.
For the situation "1-3" we are indifferent between "1" and "3" as can be seen here

```
1-3 : Show = {-0.1094, -5.2216, -0.1097}
```

which again verifies our earlier argumentation.
Finally we look at the situation "2-3"

```
2-3 : Show = {-4.6663, 0.0531, -0.1098}
```

and we see the symmetry with the situation "1-2" where we have the same expected utility of showing "2", and the same expected utility of showing "1" as we have for showing "3" in this situation.

Note that the expected utility varies with the hand type. If we get a hand of type "1-3" our expected utility of showing "1" or "3" is smaller than the expected utility of showing "2" given the hand types "1-2" and "2-3". This is reasonable since "1-3" sums to four which is the "mean" value of the game and therefore it is hard to know whether to play on "High" or "Low" at this moment when we have not seen any indication of our opponents hand. For completeness sake, we include the expected utilities for showing "1" given "1-1", "2" given "2-2" and "3" given "3-3"

```
EU(1|1-1) = 0.1118
EU(2|2-2) = -0.2215
EU(3|3-3) = 0.1116
```

and once again we see the symmetry between "1-1" and "3-3".
By looking at all these expected utilities we see that we can expect the highest winning by getting one of the extreme hand types "1-1" or "3-3" which are also the hardest to get.

MyBid

Now we turn the focus to the expected utilities for MyBid. There are too many cases to consider them all but we shall try selecting a few of them, covering some of the interesting aspects.

`Case 1 - Indifference`
As earlier mentioned, in an equilibrium of a game, the expected utility of all possible choices must be the same. As also mentioned, in this game both players have private information so we can not always count on this to be true. However, situations still occur where we have no chance of using this information as insurance against loss and in these situations our expected utilities should be the same for both our choices.
We consider the situation where we have a hand of type "1-2", have shown our opponent the "2" and she has shown us a "1". Then our expected utilities are as follows

```
MyHand : 1-2, Show : 2, OppShow : 1

MyBid = {-0.2416, -0.2414}
```

As can be seen, we are indifferent between playing "High" or "Low" in this situation which indicates that the final distributions are in an equilibrium.

If we had shown our opponent the "1" the situation would have been

```
MyHand : 1-2, Show : 1, OppShow : 1

MyBid = {-0.4973, -0.4971}
```

but we are still indifferent about the choices, even though the expected utility here is smaller due to the fact that we showed "1" in a situation where we had a "2".

`Case 2 - Symmetry`
As earlier described, the game has a symmetric nature meaning that we can expect the same situations to occur in the low and the high end.

Let us consider the case where we have a hand of type "1-2". We follow our strategy and show the "2" to our opponent, she is also showing a "2" to us. This leads to the following expected utilities

```
MyHand : 1-2, Show : 2, OppShow : 2

MyBid = {-0.4964, 0.1037}
```

We see that we have a positive expected utility by playing "Low". Due to the symmetric nature of the game, the expected utility of playing "Low" in this situation should be the same as the expected utility of playing "High" in the high end of the game.

We consider the situation where we have "2-3", have showed the "2" to our opponent and have seen a "2". Then we have the following expected utilities

```
MyHand : 2-3, Show : 2, OppShow : 2

MyBid = {0.0968, -0.5033}
```

As we can see, the expected utility of playing "High" in this situation is almost the same as the expected utility of playing "Low" in the previous situation, so this verifies the symmetric nature.

`Case 3 - Indifference and Symmetry`
Here we try looking at a case where we have a hand of type "2-2", or in other words a case that sums to four - the mean of the game. Thus, we are in a situation where our choice is completely dependent on our opponent.
First, if she is showing us a "1" the situation is as follows

```
MyHand : 2-2, Show : 2, OppShow : 1

MyBid = {0.0008, -0.4995}
```

As we can see we shall then play on "High". This seems very reasonable since the maximum sum that she can have is if her other dice is "3", and then her sum is four - the same as our own. Thus, we can never loose by playing "High".

If instead our opponent had shown us a "2" we would have this situation

```
MyHand : 2-2, Show : 2, OppShow : 2

MyBid = {-0.3996, -0.3996}
```

where we can see that we are indifferent between "High" and "Low". This is again an indication of the strategies being an equilibrium, since in the situation where we are maximum uncertain, the behavior we have seen from our opponent does that we remain indifferent between our possible choices.

And finally if our opponent had shown us a "3" we would have the following situation

```
MyHand : 2-2, Show : 2, OppShow : 3

MyBid = {-0.4998, 0.0006}
```

which is the opposite of where she showed us "1". Again a confirmation of the symmetry being kept.

`Case 4 - Extreme hand types`
As a final experiment with the expected utilities we try looking at the situation where we have got one of the extreme hands "1-1" or "3-3" - they show the same tendencies so we shall only look at "1-1".
Of course we can only show "1" so we the factor we vary is what we see from our opponent.

If she is showing us a "1" we get the following expected utilities

```
MyHand : 1-1, Show : 1, OppShow : 1

MyBid = {-0.4989, 0.0018}
```

We see that our expected utility of playing "Low" is approximately zero, indicating that we can expect a draw. This seems reasonable because the fact that our opponent showed us a "1" indicated that she must have either "1-1", which leads to a draw, or "1-3" which will make her play "High" in this situation where she has seen a "1". Thus, in both cases we can expect a draw.

If she is showing us a "2" the situation is like this

```
MyHand : 1-1, Show : 1, OppShow : 2

MyBid = {-0.8003, 0.1997}
```

We see that we have a positive expected utility on playing "Low". We know that we can never loose by playing "Low" and having "1-1", and we know that when our opponent has "1-2", shows the "2" and sees a "1", she is indifferent about playing "High" or "Low".
Furthermore, when she is showing "2" we know that she has either "1-2", "2-2" or "2-3". From the nature of the game (and the behavior we assume from a 3-sided dice, without ever having seen one in action) we know that it is twice as probable getting "1-2" or "2-3" as it is getting "2-2". So the chance of our opponent having a "1-2" is $\frac{2}{5}$, and if so, there is 50% chance that she is playing "Low". This explains that the expected utility of playing "Low" is approximately $\frac{1}{5}$.
Finally, if she is showing "3" the situation is

```
MyHand : 1-1, Show : 1, OppShow : 3

MyBid = {-0.9976, 0.0024}
```

We see that we can expect a draw by playing "Low", and are sure to loose if we are playing "High".

This concludes the examination of the expected utilities we get when using the networks we have from the intelligent agents. As promised, we now move on to examining the distance between the final distributions.

5.3.5 Distance

As a final experiment with the game "two-person high/low" we shall try observing the Euclidean distance between the distributions of OppShow and OppBid during the games. We have done this for all the other games, and seen that the distances are always going against zero.
The distance is plotted in Figure 5.7.
As can be seen the distance between the two distributions over OppShow is approaching zero, but is never reaching it during the 50.000 games.
We can also see that the distance between the two distributions over OppBid is converging, but apparently not against zero, but a value around 0.09. In the game scissor-paper-stone we saw that the distance was reaching zero,

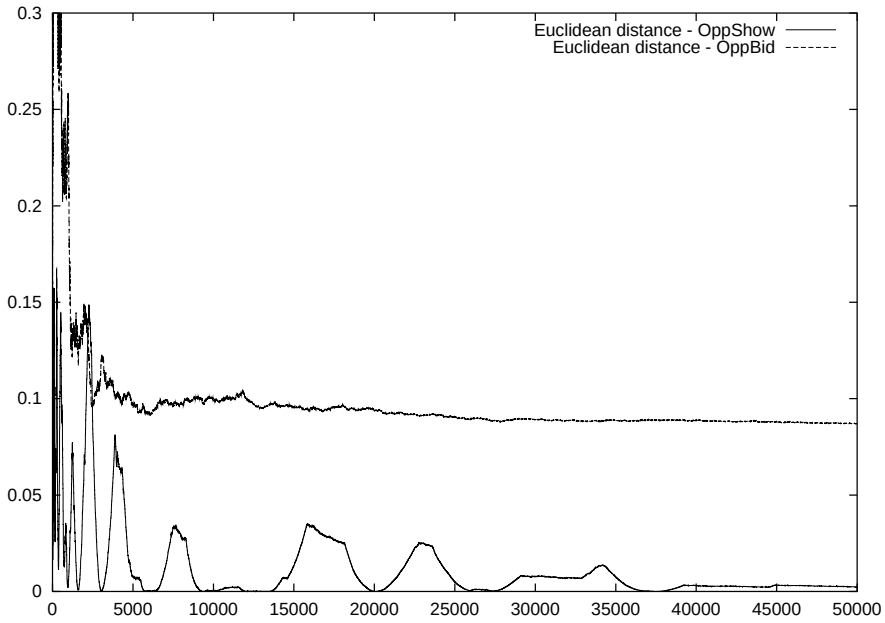


Figure 5.7: *The distance*

meaning the exact same distributions several times during the games. That it is converging against a value other than zero is an indication of the potentials being polluted. The dominated configurations that are modified in the beginning and never touched again are introducing a constant *pollution factor* that the distance cannot get below, so with this in mind it is not a problem that the distance is converging against a value larger than zero.
Furthermore, the nodes have several more states than in scissor-paper-stone, so to compensate for the number of states in the nodes when looking at the distance, we introduce a more fair distance measure for this purpose, the average Euclidean distance during the games divided by the number of states modified by the adaptation. We then get the following distances
Scissor-Paper-Stone:

$$dist_E(P1_{Opponent}, P2_{Opponent}) = 8 \times 10^{-5}$$

High/Low:

$$dist_E(P1_{OppShow}, P2_{OppShow}) = 1 \times 10^{-4}$$
$$dist_E(P1_{OppBid}, P2_{OppBid}) = 1 \times 10^{-3}$$

So we have that even though the distances in Figure 5.7 are not getting as close to zero as in the game scissor-paper-stone, we can explain why this is so, and when computing average distances, the distance between the unpolluted potentials over OppShow is getting almost as close to zero as the distance between the Opponent potentials in scissor-paper-stone.

5.4 Summary and Discussion

In this chapter we have explained the rationality of several strategies, selected to be covering the interesting cases, and we have still not found any indications that the final distributions should not be a Nash equilibrium. Furthermore, by looking at the expected utilities, we have found that the symmetry of the game is kept intact and that in the situations with maximum uncertainty, the final distributions represent randomized strategies that make random play the best strategy of an opponent.
Therefore we conclude that the final distributions are a Nash equilibrium of the game "Two-person high/low".

As mentioned in the beginning of the chapter, we were forced to use a new selection procedure in this experiment. The reason we had to do so is pollution in the dominated configurations. This is a problem because even though the configurations are dominated, the probability of them being chosen is never zero, only very close to zero. This fact combined with pollution does that the polluted values are taken into account when computing the expected utilities, meaning that a value very close to zero is added to the some of the expected utilities. Imagine that this happens in a situation with a choice between two decisions that should have the same expected utility, and one of the choices is added a polluted result and the other added an unpolluted. Then we have that the expected utility of one of the decisions is slightly larger or smaller than the other, so Brown's maximum selection criterion will result in deterministic play where a random selection was supposed to be made.

There are more than one solution to this problem. The solution we chose is to use a selection procedure similar to one of those we tested in [Jørgensen, 2000]. The idea is that as follows

- Utilities that are closer to zero than some threshold are set to zero
- If you have a choice between decisions with both positive and negative expected utilities, make a weighted selection over only the positive

- If all the expected utilities are positive, make a weighted selection over all of them
- If all the expected utilities are negative, make a weighted selection over all of them

In [Jørgensen, 2000] we showed that such a selection procedure is giving the same results as when always selecting the one decision with highest expected utility.
Another solution could be to introduce an interval around the expected utilities in which variance is said to does not matter. In this way two expected utilities varying only in this interval would be treated as equal

What we have actually shown in this chapter is that we can make a slight modification to Brown's solution procedure for finite two-person zero-sum games with one decision, making it capable of solving finite two-person zero-sum games with more than one decision.

Chapter 6

Conclusion

The intention with this thesis was to continue the work from [Jørgensen, 2000] where we made the initial experiments, indicating that it is possible to implement intelligent agents for games based on Bayesian networks. In [Jørgensen, 2000] we found that for the game of spoofing which we used as a test-bed, the intelligent agents converged against the same randomized strategies, and furthermore we were able to verify these strategies as reflecting intelligent and rational behavior.

However, we were not able to verify these strategies as being the solutions or a Nash equilibrium of the game, so one of the main purposes of this thesis was to actually prove that intelligent agents based on Bayesian networks are capable of solving any finite two-person zero-sum game.
To do so, we have looked deep into the work of George W. Brown and Julia Robinson whom have respectively suggested and proven a recursive solution procedure for finite two-person zero-sum games with one decision. We have in detail studied and described as well the procedure as the proof hereof, and even made various experiments to verify that the procedure is capable of finding the desired solutions.

The next step was the integration of this solution procedure into Bayesian networks. We found that it was similar to the training scheme, fractional updating, as also used in [Jørgensen, 2000].
We have shown that intelligent agents based on adaptive Bayesian networks are capable of finding the same solutions as Brown's recursive solution procedure. We have also verified these solutions as being Nash equilibria of the games we have used as test-beds. Furthermore, we have shown that we can extend the scheme of fractional updating with the concept of fading and still implement agents able of finding the solutions.

Thus, we have found a way to incorporate Brown's solution procedure into intelligent agents which then are not only capable of solving any finite two-person zero-sum game with one decision, these agents are also capable of exploiting potential weaknesses of their opponents. The latter follows from the fact that agents using fading are able to adapt to any context they are placed in, or in other words they are able to discover sub-optimal play of an opponent and take advantage of it. Brown's original procedure is only designed for finding optimal strategies against intelligent and rational opponents and therefore it is not capable of finding strategies that are optimal against sub-optimal play.

After having shown that we are able to incorporate Brown's solution procedure into intelligent agents that are able to find optimal strategies of finite two-person zero-sum games with one decision against any opponent they are facing, and able to find Nash equilibria of the same games by letting them face themselves, we decided to move on to more complex games.

Our motivation for doing so was that we hoped we would be able to show that our implementation of Brown's solution procedure as intelligent agents is able to solve even more complex games than the ones it is designed for. Our idea was to solve a finite two-person zero-sum games with more than one decision.
For the purpose we designed the game "two-person high/low" which not only introduces more than one decision, but also private information. Unfortunately the solution space for this game was too huge for us to compute, so the intention was to read out the suggested solutions, if any, and try verifying them as a Nash equilibrium of the game.

Due to what we defined as pollution caused by dominated strategies, we had to modify the selection criterion from Brown's solution procedure.
With the new selection criterion we showed that intelligent agents based on Brown's solution procedure integrated in Bayesian networks are converging against the same randomized strategies when set to play against each other. Furthermore, we examined the suggested solutions in detail, and to the best of our knowledge, we have verified them as being Nash equilibria of the game "two-person high/low".
Thus, we have shown that intelligent agents based on Bayesian networks are capable of solving finite two-person zero-sum games.

Chapter 7

Future Works

As the final words in this thesis, we outline some aspects that we still have not covered in our work and which we find so interesting enough to be mentioned.

All the games we have considered until now have been of the type two-person zero-sum games, so it could be interesting to see how well the ideas from this thesis apply to other kinds of games. Furthermore we have not yet considered a game without perfect hindsight which for sure would introduce some interesting problems and considerations. One could easily imagine that methods of Brown's principle will fail to solve the games as soon as perfect hindsight is not present.

Another interesting subject when talking new game types is game with more than two participants. We can easily imagine games with for example three participants where our training methods still will work correct. However, we will need to complicate our model, so problems with verification of suggested solutions will have to be taken care of, which leads to the next point.

As soon as the games get more complex we have a hard time verifying the solutions we can read out as actually being solutions. More effort has to be put into computing the exact solutions of the games we consider so we can formally verify our results.

The problems we had with the selection criterion in Brown's theorem need to be worked with. We know that we must play randomized strategies over our possible choices based on a set of weights. However, we have a problem when we have both positive and negative expected utilities at the same time, which we will have to weigh according to each other. The solutions we proposed when we discussed the problems are for sure subject to optimization.

By experiments and discussion alone, we have shown that the principles from Brown's solution procedure can be used to solve games with more than one decision. A formal proof of this result would be desired.

Bibliography

[Christiansen, 1997] Christiansen, E. (1997). Elementer af matematisk spilteori. Odense Universitet.

[Cowell et al., 1999] Cowell, R. G., Dawid, A. P., Lauritzen, S. L., and Spiegelhalter, D. J. (1999). *Probabilistic Networks and Expert Systems*. Springer-Verlag.

[Elkjær et al., 1998] Elkjær, A. S., Gundersen, B., Jeppesen, K. J., Jørgensen, T., Stausholm, M. K., Thrysøe, K., and Vestergaard, J. (1998). Dynamic Load Balancing with Bayesian Networks. Technical report, Aalborg University.

[Fraleigh and Beauregard, 1995] Fraleigh, J. B. and Beauregard, R. A. (1995). *Linear Algebra*. Addison-Wesley Publishing Company.

[Heckerman, 1995] Heckerman, D. (1995). A Tutorial on Learning With Bayesian Networks. Technical report, Microsoft Research.

[Hoehle and Kristiansen, 1999] Hoehle, M. and Kristiansen, B. (1999). Sleep Stage Classification - Learning Bayesian Classification Networks. Master's thesis, Aalborg University.

[HUGIN Expert A/S, 1998] HUGIN Expert A/S (1998). *HUGIN API 4.0 Reference Manual*.

[Jensen, 1996] Jensen, F. V. (1996). *An introduction to Bayesian Networks*. UCL Press.

[Jensen, 1999] Jensen, F. V. (1999). Gradient Descent Training of Bayesian Networks. *ECSQARU'99*.

[Jørgensen, 2000] Jørgensen, T. (2000). Adaptive Bayesian Networks for Games. Technical report, Aalborg University.

[Myerson, 1991] Myerson, R. B. (1991). *Game Theory - Analysis of Conflict.* Harvard University Press.

[Olesen et al., 1992] Olesen, K. G., Lauritzen, S. L., and Jensen, F. V. (1992). aHUGIN: A System Creating Adaptive Causal Probabilistic Networks. *Proceedings of the Eight Conference on Uncertainty in Artificial Intelligence.*

[Robinson, 1951] Robinson, J. (1951). An Iterative Method of Soving a Game. *Annals of Mathematics, Vol. 54, No. 2.*

[Russell et al., 1995] Russell, S., Binder, J., Koller, D., and Kanazawa, K. (1995). Local learning in probabilistic networks with hidden variables. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, pages 1146-1152.*

[Spiegelhalter and Lauritzen, 1990] Spiegelhalter, D. J. and Lauritzen, S. L. (1990). Sequential Updating of Conditional Probabilities on Directed Graphical Structures. *NETWORKS, 20.*

[von Neumann and Morgenstern, 1944] von Neumann, J. and Morgenstern, O. (1944). *Theory of Games and Economic Behavior.* Princeton University Press.